MACHINE LEARNING TECHNIQUES FOR DISCRIMINATION OF MENTAL ACTIVITIES

by

Kouhyar Tavakolian

BSc., Tehran Polytechnics (Iran), 2000

MSc., University of Tehran (Iran), 2003

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

MATHEMATICAL, COMPUTER AND PHYSICAL SCIENCES

(COMPUTER SCIENCE)

THE UNIVERSITY OF NORTHERN BRITISH COLUMBIA

August 2005

Your file Votre référence
ISBN: 978-0-494-28381-3
Our file Notre référence
ISBN: 978-0-494-28381-3

DEDICATION

To my family, Reza, Shamsi, Kami and Pani

Abstract

In this thesis the application of different machine learning techniques to classify mental tasks from Electroencephalograph (EEG) signals is investigated. Such classifiers are used in Brain-Computer Interface (BCI) applications. For this purpose, Bayesian graphical network, Neural Network, Bayesian quadratic, Fisher linear and Hidden Markov Model classifiers are applied to two known EEG datasets in the BCI field.

The Bayesian network classifier is used for the first time for the classification of EEG signals. Bayesian network classifier achieved more consistent compared to other classifiers.

Using classification results, a mental task can be assigned to each subject as the optimal mental task. Finding optimal mental task for subjects can have potential application in reducing the amount of training or increasing the accuracy in the BCI systems. For the preprocessing section, Independent Component Analysis (ICA) has been used.

In addition to classical Correct Classification Accuracy criteria, the Mutual Information (MI) concept is used to compare the classification results with other BCI groups.

Contents

List of Tables

List of Figures

ACKNOWLEDGMENTS

I would like to thank Dr. Siamak Rezaei, Desenka Polajnar and Dr. William Owen for their useful feedbacks on the thesis, especially Desenka for kindly offering me the opportunity to be her lab assistant.

I extend especial thanks to Dr. Saif Al Zahir, Dr. Charles Brown, Dr. Liang Chen and Dr. Kenneth Prkachin for what I have learnt from them in their courses and in particular Dr. Karim Arabi for his guidance and helps during the past two years.

I would like to thank my friends in Prince George, Melani Thomas, Clayton Dyck, Eric Tompkins, Noushin Naziripour, Farnaz Behzadmehr and Adrian Batho. I should also thank Farsheed Vala, Joanne Bai and Robyn Dyck for their helps in proof reading and my roommates Kiarash Naziripour, Maki Umeda and Pruthvi Polam. I want to thank my aunt Fakhri and my uncles, Hossain and Abdollah Rajab for their support.

I want to thank Akbar Ganji the Iranian journalist who is on the food strike now for the 60th day. He is doing this for the sake of freedom of people of my country, Iran, from tyranny and theocracy.

Lastly but most importantly I can not thank enough my God, my father and my mother without whom my existence is meaningless. Their love and support will always be a driving force behind me.

Chapter 1

Introduction

1.1 Context

During the past two decades many research groups have focused on the development of brain computer interface (BCI) systems. "A BCI is a communication system that allows a subject to act on his environment solely by the means of his/her thoughts, without using the brain's normal output pathways of muscles or peripheral nerves" [1].

Ideally, the BCI should provide the user with an alternative method for acting on the world by performing few mental activities that do not need any overt physical movement. People with motor disabilities[1] can benefit from such a system by using specific brain activations to communicate via a computer.

The electrical current generated by nerves firing in the brain diffuse through the head. The produced electrical current can be measured by metal electrodes placed on the scalp. This signal is called the electroencephalogram or EEG. Most BCI systems have been based on the EEG signal because it has the advantages of good time resolution, being inexpensive and noninvasive.

[1]such as amyotrophic lateral sclerosis (ALS) patients

The EEG gives a coarse view of neural activity and for decades has been used to non-invasively study the physiology of the brain. Over the past twenty years, many research studies have demonstrated a correlation between EEG signals and mental activities. In addition, the rapid development of personal computers that enables them to be used in real time processing of multi-channel EEG has provided another main reason for scientists to focus on the development of brain computer interface systems.

Using a BCI subject can control a device, such as cursor on the screen, by performing mental activities that are associated with actions and are dependent on the BCI application. Typical BCI applications include control of the elements in a computer environment (e.g. cursor movement), command of an external device (e.g. prosthesis, robot), spelling programs (e.g. virtual keyboard) and sometimes even computer games such as PACMAN.

Figure 1.1 shows the block diagram of the processing part of a BCI system. The EEG signal first goes through a pre-processing stage. In this stage the effect of artifacts (e.g. eye movement) or noises are removed from the signal. In the next stage proper features are extracted from this signal. These features are vectors of numbers that are assigned to each individual EEG window.

The classifier then uses the extracted feature vectors to train itself. After training, the classifier becomes able to classify inputs. The main focus of this thesis is on the pre-processing and the classifier stages[2]. The feature extraction stage in this thesis will be mainly based on the past works [4].

Considering its non-stationary and chaotic nature, the analysis of EEG data and the extraction of information from it is a challenging signal processing and pattern recognition problem. For this reason, despite the technological developments, numer-

[2]for more information about BCI block diagram refer to chapter two.

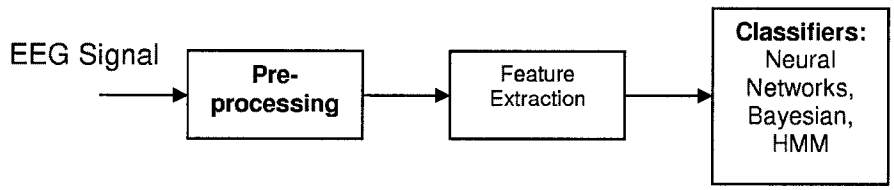

Figure 1.1: General Block Diagram of BCI systems

ous problems still exist in building efficient BCIs. The biggest challenges are related to *accuracy, speed and usability.*

In this thesis the focus is on the accuracy problem of BCI systems by developing and comparing different machine learning techniques for the *classifier stage.* To do so, these machine learning techniques are applied to two known EEG datasets in the BCI field. Both of these datasets are available online on the internet [3] [35].

1.2 Past Work

In past research, the classification of mental tasks using the Purdue University EEG dataset [2] and the dataset gathered at the Research Center of Intelligent Signal Processing (RCISP) in Tehran-Iran was investigated.

The RCISP EEG dataset has been gathered from five subjects during the performance of three mental tasks, rotation, multiplication and baseline, using a nineteen channel EEG system. Feedforward neural networks were used as the classifier in past research [4].

As can be seen in Figure 1.1, for the first module, the EEG signal is the input. Past experience has shown [5] that for every subject and set of mental tasks, some EEG channels are superior to others in terms of classification. In past research, genetic algorithm was proposed as the selection criteria of these superior EEG channels in classification of mental tasks. So, an algorithm was proposed to reduce nineteen EEG channels to six channels from a dataset taken from the EEG instrument in RCISP

[6].

After recording, the EEG signal should be pre-processed to remove, or at least reduce, background activities. The eye blink is one of the background activities with the worst effect on the EEG quality. To remove this artifact, a special time domain filter was implemented [4].

Afterwards proper features should be extracted from the EEG signal. These features can address different aspects of the signal such as time domain, frequency domain, statistical, nonlinear, and chaotic characteristics [4]. All these features have been extracted and compared in the previous research. Some of these features include power spectral density, AR coefficients, time domain statistics, and fractal dimensions. Fractal dimensions were calculated by Higuchi and Petrosian methods, which could be calculated much faster than other features.

The efficiency of the two hybrid feature vectors in mental task classification was also investigated in past research. These vectors were composed of both linear and nonlinear features. According to the achieved results from hybrid feature vectors, a genetic algorithm was developed to find the optimum combination of different features, in classification of mental tasks.

Five mental tasks were classified from the total signal of two sessions. The average classification accuracy for four subjects on the Purdue dataset was 71.85% and for the case where three sessions of EEG was used, for one of the subjects, the result was 70.42%. The window length in this case was one second with half a second of overlap.

1.3 Current Work

In the present research, the classification of mental tasks using the Purdue University EEG dataset [2] and the EEG dataset given by A. Schlogl from Department of Medical Informatics, University of Technology Graz [3] are investigated. Both datasets are

known and well established datasets in the BCI field and they will be explained in chapter four.

The current research mostly focuses on the classifier stage (as can be seen in Figure 1.1). Also, as the Purdue University dataset had five different mental tasks, it was investigated for finding the best mental tasks for each subject.

The previous pre-processing stage, mentioned in section 1.2 has also been improved by using Independent Component Analysis (ICA) [7]. ICA was used previously by other researchers to remove artifacts from EEG as well [34].

For the current research, AR coefficients and AAR coefficients were extracted from the EEG windows for all classifiers. These extracted features will be inputted to the next stage, which is the classifier. The same extracted features for all classifiers, facilitated the comparison of classifiers' efficiencies.

In general, different machine learning algorithms have been applied to the EEG signal as the classifier and the results are compared with each other. The main focus was on investigation and comparison of different classifiers in classification of mental tasks. This comparison included feedforward neural network, Bayesian quadratic, Bayesian network, Fisher linear classifier and Hidden Markov Models (HMM). These classifiers are known methods in the machine learning literature as will be explained in chapter three.

Considering the Matlab [8] software capabilities and its toolboxes for signal processing, it was used for the implementation of the algorithms. The neural network, signal processing, Bayesian network, HMM and pattern recognition toolboxes of Matlab were used in this thesis [9].

1.4 Organization of The Thesis

In chapter two, the structure of Brain Computer Interface systems will be investigated and previous work in the field will be explained. First, a brief introduction of the EEG characteristics, origin, and its relation to BCI systems is presented. An explanation of the structure of Brain Computer Interface systems in terms of its block diagrams is then presented. There will also be a comprehensive literature review of past works in the BCI field. EEG measurement techniques and standards are addressed at the end.

In chapter three the machine learning and signal processing techniques that were used in implementing the algorithms are investigated. These algorithms include neural network, Bayesian quadratic classifier, Bayesian network, Fisher linear classifier, HMM, Data splitting techniques, ICA, and our feature extraction techniques, which are AR and AAR coefficients.

In chapter four, results for different classifiers will be presented and compared with each other. In this chapter two EEG datasets are introduced. These datasets are known datasets in BCI field. ICA is also applied to these datasets as a preprocessing block. In the appendix some of the Matlab program codes are presented.

In chapter five there will be a conclusion regarding the results and possible future work that can be based on this thesis and its contributions.

Chapter 2

Brain Computer Interface Systems

2.1 Introduction

In this chapter the background and the structure of brain computer interface systems will be investigated and previous works in the same field will be considered. Given that the focus here is on EEG-based BCIs, it is prudent to review the characteristics of EEGs and how these characteristics relate to BCI systems. This section then leads into an explanation of EEG standards and measurements.

The structure of brain computer interface systems will be examined using block diagrams in the third section of this chapter. At the end there will be a comprehensive literature review of past work in the field. Five more important and inspiring projects in the BCI field are investigated, including Graz, Wadsworth, Adaptive Brain Interface (ABI), Berlin BCI (BBCI), Thought Translation Device (TTD) and Neil Squire Foundation.

2.2 Electroencephalography (EEG)

Electrobiological measurement involve measuring small electrical activities associated with biological systems. Electrobiological measurements include electrocardiography (ECG, heart), electroencephalography (EEG, brain), electromyography (EMG, muscular system), magnetoencephalography (MEG, brain), electrogastrography (EGG, stomach) and electrooptigraphy (EOG, eye dipole field). Computer tomography (CT), magnetic resonance imaging (MRI), functional MRI (fMRI) and positron emission tomography (PET) are other imaging techniques that are based on different physical principles.

At present, MEG and fMRI systems are expensive and require magnetically shielded environments. Being dependent on blood flow, fMRI and PET have poor time resolutions. So, electrical potential measurements are a more practical choice to monitor brain activities for applications like BCI [10].

"The EEG is the electrical activity of an alternating nature recorded from the scalp surface after it is being picked up by metal electrodes" [10]. The brain electrical currents that are measured directly from the cortical surface of the brain is called an electrocortiogram and it is a highly invasive procedure. It is evident that signal to noise ratio (SNR) can be increased substantially by invasive technologies such as electrocortiograms. Research in the electrocortiogram field is limited because normal people may be reluctant to be subjects in such experiments.

In comparison, EEG reading is a completely non-invasive procedure that can be applied to patients, children and normal adults with almost no risk or limitations. EEG also has a high time resolution, simple acquisition scheme, and is inexpensive. All these advantages have made the EEG the preferred tool in the BCI field.

2.2.1 History and Applications of EEG

The existence of electrical currents in the brain was discovered in 1875 by English scientist, Richard Caton. He observed electrical activities from exposed brains of rabbits and monkeys [10]. In 1924, Hans Berger, a German neurologist, used Einthoven's string galvanometer to record the electrical activity of the brain in human subjects. He declared that weak electric currents generated in the brain can be recorded without opening the skull, and also can be depicted graphically on a strip of paper.

The activity that he observed changed according to the functional status of the brain, such as in sleep, anesthesia and in certain neural diseases. He named this activity as Electroencephalography. These findings made the foundations for many of the present applications of electroencephalography. Later on in 1934, regular oscillations (10 to 12 Hz) were discovered by other scientists and were termed alpha rhythm [10].

One of the major roles of EEG is in the diagnosis of epilepsy. Abnormal patterns such as spikes, sharp waves and wave complexes can be used for this diagnosis. The type of EEG activity and the area of the brain from which it is recorded would assist the physician in prescribing the correct medication for that particular type of epilepsy. Sometimes patients with epilepsy which cannot be controlled by medication have surgery to remove the damaged tissue. The EEG plays an important role in localizing these damaged tissues. These EEG recordings can be carried out for periods ranging from a day to a week. The recorded EEG can roughly show that areas of the brain that should be surgically removed.

EEG studies can also be used in patients who are deeply unconscious to discriminate between brain death and possible reversible conditions. Electrocerebral inactivity (ECI) is defined as no EEG activity more than 2 micro volts in amplitude when recording from electrodes on the scalp that are at least ten centimeters apart from each other.

EEG has extensive applications in biofeedback. Biofeedback is defined as a method for learning to increase one's ability to control biological responses, such as blood pressure, muscle tension, and heart rate. Different methods such as EEG, ECG, skin temperature, skin conductance and EMG have been used to measure physiological responses and make them apparent to the patient, who then tries to alter and finally control them without the aid of the monitoring devices.

Previously, self regulation of the EEG alpha rhythm was used for biofeedback and for relaxation purposes. Now it is used in the treatment of numerous mental disorders such as Attention Deficit Hyperactive Disorder (ADHD), panic attacks and sleep disorders [5].

The site of action of many anesthetic drugs is the brain. With increasing depth of anesthesia, there is a progressive increase in signal amplitude and decrease in frequency. Therefore, it seems to be reasonable to consider the EEG as the principal signal to assess the depth of anesthesia. There are many studies that have applied different signal processing techniques to EEG to assess anesthesia depth [11].

EEG has been used to investigate sleep disorders and physiology. For example, a one channel EEG signal has been used to develop an automatic sleep stage scoring system [12]. Genetic algorithm and neural network were used to develop this system.

A new area of research in EEG is brain fingerprinting. An EEG-based brain-state analysis system was developed as a potential tool in the fight against terrorism. This system is mainly based on detection of the P300 signal in the EEG to detect possible criminals. P300 is one of the brain's evoked potentials that has been used in the BCI field too. It is a specific, electrical brain wave response that is emitted by the brain within a fraction of a second when an individual recognizes an incoming stimulus that is significant [13].

Finally, EEG-based BCI is one of the more recent applications of EEG that is the

main topic of this thesis and will be discussed in the rest of this chapter.

2.2.2 EEG Origin and Characteristics

"EEG measures the current that flows during synaptic excitations of the dendrites of many pyramidal neurons in the cerebral cortex" [10]. Brain electrical current consists mostly of Na+, K+ ions that are pumped through channels into neuron membrane.

A large population of active neurons can generate electrical activity that penetrates through skin, skull, and several other layers. These weak electrical signals detected by the scalp electrodes are amplified, pre-processed, and stored to computer memory [10].

The temporal resolution of EEG is in the millisecond range, which is significant compared to other measurement techniques. The EEG spatial resolution is in the centimeter range whereas in MEG, PET or fMRI are in the millimeter range. The amplitudes of the EEG signals are typically between 2 to 100 micro volts. The bandwidth range of the scalp EEG is up to 100 Hz, where the major power is distributed in the range of 0.5 to 60 Hz.

It is known that different cortical areas in the brain have distinct functions that are related to different mental tasks. A few BCIs, including the ABI, are based on classification of different spontaneous mental tasks [5]. These tasks should activate certain cortical areas and produce different EEG rhythms. Keirn [2] from Purdue University and Anderson [15] in Colorado State University were the first to study the possibility of using classification of mental tasks as a basis of BCI. In Figure 2.1, a mapping of different cortical areas in the brain can be seen.

There are various characteristics of EEG signal that can be used as a basis for a BCI system. Event-related potentials (ERPs) and rhythmic brain activity are two of these characteristics which are explained in the next sections.

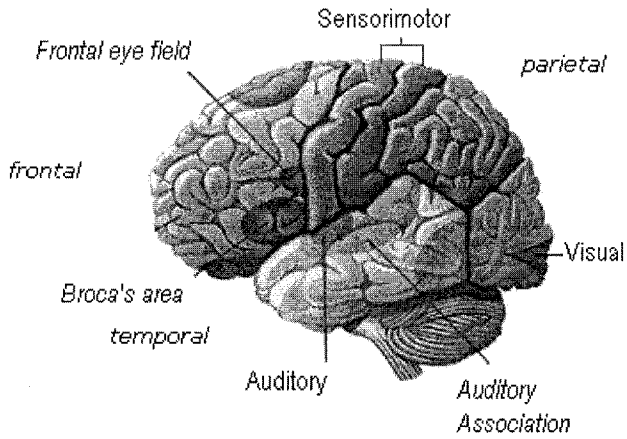

Figure 2.1: Cortical areas of brain [5]

2.2.3 Event-Related Potentials

ERPs are the potential changes in the EEG that occur in response to a particular stimulus or event. Sometimes these changes are so small that in order to be detected, EEG samples must be averaged over many repetitions. This averaging can remove the random fluctuations of the EEG which are not related to mental activities.

One of the commonly studied ERPs is the P300, which is a positive deflection in the EEG. This positive deflection occurs around 300 ms after stimulus onset. To produce the P300, the subject is exposed to a rare stimulus, which occurs randomly between other frequent stimulus. When a frequent stimulus is encountered, it is seen as being insignificant so a P300 is not produced. Donchin and Smyth used the P300 to implement a BCI system [17].

Slow cortical potentials (SCPs) are potential shifts of the EEG lasting from several hundred milliseconds up to several seconds. They are in 1-2 Hz frequency range. SCPs are detectable in every human brain, even in patients whose motor periphery is completely disconnected from the central nervous system. A research group in Tubingen university in Germany used SCP to develop a BCI called thought translation device. Their work will be explained later in section 2.5. Evoked potentials are

those ERPs that arise in response to a certain physical stimulus[1]. A typical evoked potential is the visual evoked potential (VEP) that demonstrates the output features of the visual pathway. The EEG over the visual cortex varies at the same frequency as the stimulating light.

Sutter, at the Smith-Kettlewell Eye Research Institute in San Francisco USA, used VEP as a tool to interface the computer. The subjects had scalp electrodes placed over their visual cortex. He presented a 64-position block on a computer screen. By processing the EEG, with a high accuracy he detected which block the subject was looking at. Each of these positions was assigned to a symbol such as a alphabet letter or an English word [16].

Sutter used a lengthy binary sequence to switch 64 symbols between red and green. Each symbol was included in several subgroups and the entire set of subgroups are presented several times. Each subgroup's VEP amplitude were calculated about 100 ms after the stimulus and was compared to a VEP template established for the user. From these comparisons, the computer detected with high accuracy, the symbol that the user was looking at [17].

2.2.4 Rhythmic Brain Activity

In different levels of consciousness, the brain waves in an average person's brain show certain rhythmic activity. For example, the different sleep stages can be detected in the EEG. These rhythms are affected by actions and thoughts. For example, the planning of a movement can block or attenuate a particular rhythm called mu. The fact that thoughts affect the brain rhythms has been used as the basis for BCI systems [14] [5].

The EEG signal can be divided into several frequency ranges. They are named

[1]visual, auditory or somatosensory

by Greek letters Delta, Theta, Alpha, Beta and Mu. These ranges set the limits in which the different brain rhythms can be observed.

Delta rhythm The part of EEG spectrum that occupies 0.5-4 Hz belongs to the delta waves. Delta waves appear in infants, deep sleep and in some brain diseases. Therefore these waves are not useful for BCIs.

Theta rhythm The part of EEG spectrum that occupies 4-8 Hz belongs to the Theta Waves. Theta rhythm plays an important role in infancy and childhood. In normal adults theta waves are seen mostly in drowsiness and sleep. During waking hours the EEG contains only a small amount of theta activity and there is no organized theta rhythm. This activity occurs mainly in the temporal and central areas.

Alpha rhythm This rhythm, at 8-13 Hz, occurs during wakefulness over the posterior regions of scalp, generally with higher voltage over the occipital areas. The amplitude is variable but is mostly below 50 micro volts in adults. It can best be detected when eyes are closed and under conditions of physical relaxation and relative mental inactivity. It is blocked or significantly decreased by attention, especially visual, and mental efforts. [5].

Beta rhythms Any rhythmical activity in the frequency band of 13-30 Hz is regarded as a beta rhythm. Beta rhythm amplitudes are seldom larger than 30 micro volts. Beta rhythms can mainly be found over the frontal and central region. A central beta rhythm is related to the mu rhythm that follows next.

Mu rhythm In awake people, primary sensory or motor cortical areas often display 8-12 Hz EEG activity when they are not engaged in processing sensory input or producing motor output. This activity is called mu rhythm when focused over the motor cortex. Although the frequency and amplitude of the mu rhythm are

similar to alpha rhythm, the mu rhythm is topographically and physiologically different from the alpha rhythm [14].

Research has shown that movement or preparation for movement is typically accompanied by a decrease in the mu and central beta rhythms, especially contra-lateral to the imagined movement [18]. This decrease has been called Event Related Desynchronization or ERD. ERDs do not require actual movements. This makes them a suitable choice for BCIs.

An example of ERD-maps [5] calculated during imagination of right and left hand movements is shown in Figure 2.2. The focus of activation is localized contra-lateral to the side of motor imagery.

The Mu rhythm has been used extensively in implementation of BCI systems. The BCIs developed in Wadsworth Center and Graz University are two of these type of BCI's that will be explained in section 2.5.

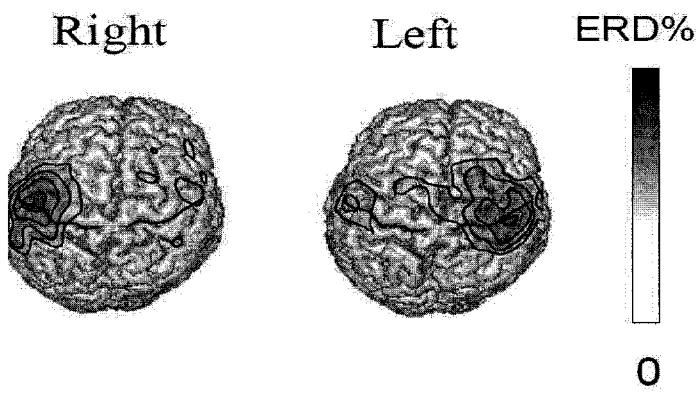

Figure 2.2: ERD maps for right and left imageries [18]

2.3 EEG Measurement

Since 1958, the International Federation of Electroencephalography and Clinical Neurophysiology has adopted standards for electrode placement which is called the 10-20 electrode placement system. This system has standardized physical placement and designations of electrodes on the scalp. This is accomplished by the head being divided into proportional distances from some skull landmarks, nasion, pre-auricular points and inion, to provide adequate coverage of all regions of the brain.

The name 10-20 designates proportional distance in percentage between ears and nose where points for electrodes are chosen. Electrode placements are labeled according to underlying brain areas: F (frontal),T (temporal), P (posterior), C (central) and O (occipital).

The letters are accompanied by odd numbers at the left side of the head and with even numbers on the right side. Left and right side is defined by convention from point of view of a subject. In both EEG datasets used in this thesis the 10-20 standard has been used.

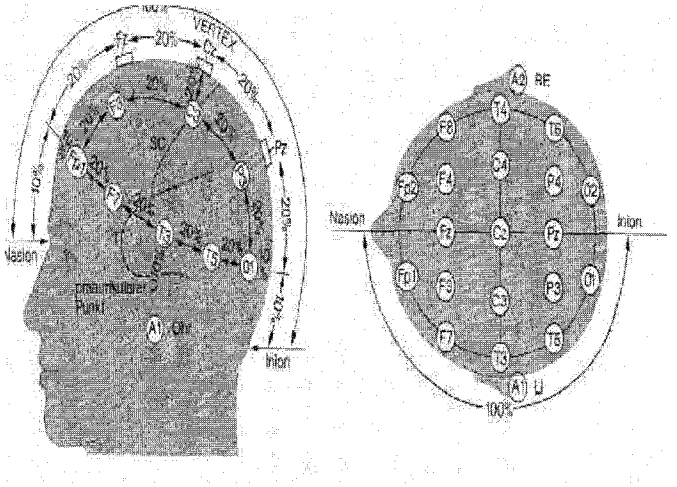

Figure 2.3: 10-20 system of EEG electrode placement [10]

2.4 BCI Definition and Background

In this section the brain computer interface is defined and then the block diagram of a typical brain-computer interface system will be explained.

2.4.1 BCI Definition

In the first international meeting of BCI researchers the brain-computer interface has been officially defined as "a communication system that does not depend on the brain's normal output pathways of peripheral nerves and muscles" [1]."The pattern of changes in the EEG reflects some large-scale brain activity that makes it suitable as a communication tool" [5] so as stated earlier, most BCIs developed so far are based on the EEG signal.

It was in the 1990's when BCI research started. Faster computers and better EEG devices offered new possibilities to process the EEG signal. To date, there have been many BCI research groups all over the world. They have taken different approaches to the problem. Not all of these BCI groups developed an online BCI system that can give feedback to the subject [5]. Except for the Graz [3] group, none of the BCIs have yet become commercially available.

Despite the technological developments, there are still challenges in reaching the ideal BCI. This chalenges include the accuracy, speed and usability of BCI. Considering the current technology, some other interfaces are still more efficient compared to EEG-based BCIs. If, for example, the patient can move even one of his/her muscles in a controlled way, the interfaces based on EMG should be more efficient compared to EEG-based BCIs.

However, BCI could provide a new communication tool for patients suffering from ALS. They are completely paralyzed physically and unable to speak, but they are cognitively alert.

2.4.2 Block Diagram of BCI

Figure 2.4 shows a block diagram of a brain-computer interface system.

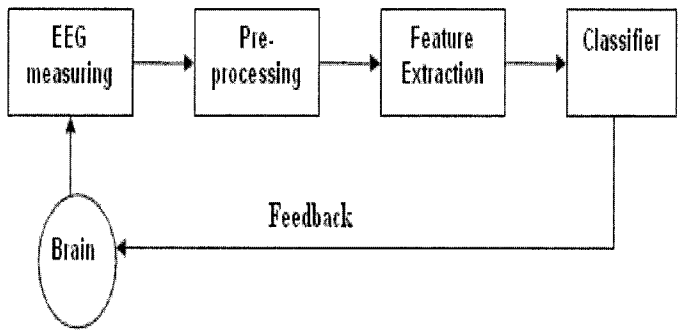

Figure 2.4: Block Diagram of a BCI system

a) EEG Measurement This measurement is done by using the electrodes that are typically placed according to the 10-20 standard. Firstly, the EEG signal is converted to digital by an A/D converter. The initial filtering of the signal is done in the EEG instrument. For BCI applications, EEG is measured and sampled while the user performs different mental activities.

b) Preprocessing This processing includes artifact removal and initial filtering of the EEG signal. One of the artifacts that significantly affects BCI performance is the eye blink and there are different algorithms for removing it. One of these algorithms is implemented in this thesis and will be explained in chapter three.

c) Feature extraction At this stage, certain features are extracted from the digitized and preprocessed EEG signal. In a simple form, a certain frequency range is selected and its amplitude is measured. Regardless of the nature of features, it is desirable to have a distinct set of features for each mental task. If extracted features have less overlap and are distinct enough they can be classified more accurately.

d) Classifier The features extracted in the previous stage are the input for the

classifier section. The classifier, ranged from a simple linear discriminant model to a complex nonlinear neural network or HMM that can be trained to recognize different mental tasks. The classifier should calculate the probabilities for the input belonging to each class. Usually the class with the highest probability is chosen as the output. It is then possible to detect the task-specific EEG patterns from the EEG samples with a certain level of accuracy.

e) Feedback The classifier's output can simply be transformed to a particular action. The action can be a sound or a movement of the cursor on the screen. This action can be shown to the subject as a feedback.

2.5 BCI Systems

In this section six important research works in the BCI field will be reviewed. All of these groups have ended up developing a real time BCI system though there are many other groups working in the same field [5].

2.5.1 The Wadsworth Center

With the BCI system of Wolpaw et al [18], subjects learn to control mu or central beta-rhythm amplitude and use it to move a cursor in one or two dimensions on a screen. For example, the user increases the amplitude of 8-12 Hz mu rhythm to move a cursor towards a target at the top of the screen, or decrease it to move a target towards the bottom of the screen [17].

Spectral amplitude was calculated from EEG data using the autoregressive coefficints (AR). In its simplest case, the amplitude of a single spectral band at a single location on the scalp was used to move the cursor in a one-dimensional path (1-D). Gradually, by providing feedback, the user could learn to control this amplitude. In

other cases, for each dimension of cursor control, a linear equation translates mu or beta rhythm amplitude from one or more scalp locations into cursor movements. The cursor movement is updated 10 times per second [18].

In one study [17], the user learned over a series of 40 minute training sessions, to control cursor movements. Significant control was finally acquired within two to three weeks. In the initial sessions, motor imagery was used to control the cursor. As training proceeded, imagery usually became less important, and users moved the cursor like they perform normal motor acts and without thinking about the details of performance. By this, users could move the cursor to answer yes/no questions with accuracies more than 95%. They could also achieve independent control of two different mu or beta rhythm channels and use that control to move a cursor in two dimensions.

Previously, cursor movements occurred in one dimension as a linear function of the EEG features. In another study, three different methods for mapping of the classification results on the computer screen were evaluated [19]. For each method, a cursor function for possible online application was evaluated. The methods differ in the dimensionality of the cursor movements and whether the cursor function is linear or nonlinear. The first method is 1-D linear, the second method is two-dimensional (2-D) linear and the third one is 1-D nonlinear. According to the offline analysis, they have claimed that the other two methods have performed significantly better than a linear 1-D cursor function.

In past work Wolpaw et al, extracted one or two features from the EEG signal. In a current study up to 30 EEG extracted features were used. The EEG signal was taken from five trained users [20]. Performance improved with more features up to a point and then decreased slightly. By averaging over the users, maximum performance could be obtained with ten to twenty features. These results suggested that online

BCI performance can be improved by using more of the information available in the EEG signal.

In a recent study, Wolpaw et al, have examined the presence and characteristics of EMG contamination during initial BCI training sessions [20]. Their present data comprise the first ten sessions of BCI training from seven users. Based on their results, they have concluded that EMG contamination arising from cranial muscles is often present early in BCI training and gradually disappears. In those users who eventually acquired EEG control, early target-related EMG contamination may be most prominent for unsuccessful trials and can reflect user frustration. In those users who never acquired EEG control, EMG can initially serve to move the cursor toward the target.

It was also concluded that comprehensive topographical and spectral analyses throughout user training are essential for detecting EMG contamination and differentiating between cursor control provided by EEG control and cursor control provided by EMG contamination [20].

2.5.2 Graz BCI

This BCI was developed in Department of Medical Informatics, University of Technology Graz. Like Wadsworth-BCI, the Graz-BCI also works with mu and beta-rhythm to move a cursor on a screen. This group also developed the system to drive a neuroprosthetic device, or even the natural muscles by functional electrical stimulation or FES. The Graz-BCI works with the users producing images of movement in their minds. The Graz-group has also developed a remote control telemedicine system for the training sessions to be performed from home.

One of their research topics [21] is to investigate the possibility of classifying EEG data for on-line BCI operation by using both a recursive least squares (RLS)

algorithm to estimate adaptive autoregressive coefficints (AAR) coefficients and a linear discriminant analysis (LDA) algorithm as the classifier. In comparison to band power estimations and neural network-based classifiers, a combination of RLS and LDA methods in off-line analysis has been found to improve classification accuracy.

To get this result the Graz group have had four subjects, and EEG signals were analyzed in subject-specific frequency bands and classified on-line by a neural network. The neural network output was used as a feedback signal. The on-line error was between 10% and 38.1%. On the other hand, the single-trial data were also analyzed off-line by using an adaptive autoregressive (AAR) model of order 6. With a linear discriminant analysis the estimated parameters for left and right motor imagery were classified. The error rate obtained varied between 5.8% and 32.8% and was, on average, better than the on-line results obtained by the neural network.

In the BCI previously used in Graz studies since 1990, the experimental paradigms were programmed in C and installed under MS-DOS. The system used a digital signal processor (DSP) board in addition to a PC.

This dependency on a DSP board increased the complexity of the BCI system, because DSP boards are usually programmed in C or assembly languages and therefore are harder to maintain or expand.

In the newer BCI they removed the DSP board and by installing the real time Kernel for windows they can directly analyze signals in the Matlab environment. The system could read and classify EEG data sample by sample with a maximum rate of 66 kHz without buffering or using the DSP board.

Another advantage of the new Graz-BCI is that it can be remotely controlled over internet connection. Acceptable results were achieved on three male subjects using this new BCI [22].

2.5.3 Thought Translation Device

In many years of research, Niels Birbaumer and his colleagues from the University of Tubingen in Germany have shown that people can control their Slow Cortical Potentials or SCPs [23] [24].

They have developed software called Thought Translation Device or TTD that consists of a computer program which reads data from an EEG-amplifier system, then performs on-line processing and provides feedback of the processed signal. TTDs with auditory or even tactile feedback are also available.

After a long training period, users switch to a language support program that can give certain communication capabilities. A spelling program included in the TTD allows subjects to select single letters by sequential selection of blocks of letters presented in a dichotomy structure with five levels. In order to improve the speed of the communication, this program was supplemented by a dictionary offering word completion after only a few letters were selected by the subject.

By using TTD, eleven completely paralyzed ALS patients were able to write messages or letters of considerable length. A special internet browser had also been developed for the TTD that allowed subjects to access the internet by selecting links with their brain responses [25].

The SCPs amplitude shifts are referenced to the final SCP value of the 2-s preparatory phase immediately before the feedback starts [25]. SCPs are calculated by a moving average over 500 ms of EEG activity that is updated every 62.5 ms. At the end of the feedback phase, the SCP shift is classified as a negative or positive response according to an algorithm that calculates the integral of the SCP shift across the feedback period. The classification methods for SCPs are described in [24].

2.5.4 BBCI

The Berlin brain computer interface (BBCI) is an independent, EEG-based BCI that has been recently developed. It works with 128 EEG electrodes. BBCI records differences in the so called readiness potentials, variations of beta wave-patterns and differences between the right and left brain hemispheres (asymmetric patterns), depending on which hand is used for movement. The big advantage of this approach is that there is no need of a lengthy training for the user.

Initially, the main interest of the BBCI group was the development of multimedia applications like the PACMAN-game. About 20 minutes of machine training are required to play PACMAN by thoughts with the BBCI. In fact, unlike the SCP- or mu-based BCIs, with the BBCI it is not the user who learns but rather it is the machine that learns. This BCI is accomplished by the complex signal processing techniques applied to EEG.

For machine training the user should sit comfortably. It is important to avoid disturbing muscular artifacts like biting, yawning and to minimize eye movements. The actual training is performed in three to four sessions of seven minutes each. During these sessions, the users have to use a keyboard with their right and left hand, respectively and the different levels of EEG-activity are recorded. All data are sampled and a movement-related potential is extracted to drive a cursor on a screen.

In a recent work, the BBCI group have tried to enhance bit transfer rates in BCI by using feature combination and multi-class paradigms. This work includes new algorithmic aspects such as the development of new feature combination strategies and a new algorithm that fully generalizes their previous work on the Common Spatial Pattern [26].

2.5.5 Neil Squire Foundation

During the past ten years, Gary Birch and his colleagues in the Neil Squire Foundation, have developed a BCI system. Their BCI is mainly based on methods to detect user-generated patterns in the user's EEG, related to imagined movements [27].

Before the year 2002 they had developed a single-position, brain-controlled switch that responds to specific patterns detected in spatiotemporal EEG measured from the human scalp. This design was called the Low-Frequency Asynchronous Switch Design (LF-ASD). They had considered asynchronous applications that work when the user intends to control but also remains in a stable off state when there is no intent to control.

The usability of the LF-ASD was evaluated on a large subject population, which included normal and high-level spinal-cord injured subjects [28]. For example in one study, two male subjects with high-level spinal cord injuries were selected and asked to imagine a right-hand index finger flexion. In this study the LF-ASD performed a sample-by-sample classification of each feature vector every 1/16 of a second using a 1-Nearest Neighbor algorithm as the classifier [29]. In this study, it was concluded that spinal cord-injured subjects could operate these new designs to the same ability as normal subjects.

2.5.6 Adaptive Brain Interface Project

The Adaptive Brain Interface (ABI) has been developed under the project "Adaptive Brain Interfaces" financed by the European Commission which ended in 2001 [5] [30].

The ABI was able to classify three mental tasks from online spontaneous EEG signals with around 70% accuracy. Classification decisions were made in every 0.5 s window. Also the training time required to achieve this level of performance has been short; only a few days of moderate training.

The power spectrum density components in the frequency band of 8-30 Hz were extracted as features within 1/2 second EEG segments. Classification was done using a classifier called local neural classifier [31]. In this classifier, each mental task is represented by a prototype in a high dimensional input space.

The aim was to find the appropriate position of the prototypes in this space to be able to differentiate the classes. So, during the training process, the prototypes were pulled towards the EEG samples of the mental task they present and were pushed away from the EEG samples of other mental tasks.

The mental tasks used in the ABI developed in Helsinki University were: relax, subtractions, cube rotation, word association and right or left hand movement. In the word association task the subject forms successive words in his or her mind in such a way that the next word starts with the last letter of the previous word. Eyes remain open in all the other mental tasks except in the relax task.

In a more recent work, researchers from the same ABI project have shown how volunteers could, within a few days, learn how to master a portable EEG-based brain-computer interface that recognized three mental states [32]. Two participants successfully moved a robot between several rooms by mental control only.

As a second demonstration of brain-actuated interaction, they have described a communication tool that enables people to select letters from a virtual keyboard and write messages. In Table 2.1 there is a summary of the work done by the BCI groups explained previously [33].

Group	Mental Activities	Electrodes Features Recognition algorithm	Application Number of subjects
Wadsworth center USA	Mu and beta rhythm modulation	-64 EEG electrodes -Power in the mu and beta band -Linear classifier	-Synchronous - 1 and 2 dimensional positioning of a cursor on screen -Eight subjects
University of Tübingen Germany	Control of slow cortical potentials	Fz, Pz, Cz Low-pass filtering Thresholding	-Synchronous -on/off switch -Eleven locked-in patients
Technical University of Graz Austria	Imagination of left and right hand, and foot movements	- 2 electrodes 2.5 cm -C3 and C4 - Power in the alpha and beta band and AR coefficients -Linear discriminant analysis and hidden Markov models	-Synchronous -virtual keyboard, hand orthosis control and cursor movement
Berlin (BBCI)	Readiness potentials	-128 Electrodes -combination features such as CSP and AR coefficients - LDA	-Asynchronous -Packman video game -Ten subjects
Neil Squire Foundation Canada	Recognition of movement imagination against other MAs	- Bipolar recordings F1-FC1, Fz-FCz, F2-FC2, FC1-C1, FCz-Cz Bi-scale wavelength analysis - 1-Nearest neighbor classifier	-Asynchronous switch -Seven subjects -51 bits/min (max)
ABI Project European Union	-Relax, imagination of left and right hand movement, cube rotation, subtraction and word association.	-F3, F4, C3, Cz, C4, P3, Pz, P4 -Power in 2 Hz wide bands from 8 to 30 Hz -Neural network	- Asynchronous control of a mobile robot - Five subjects

Table 1 Summary of previous work in BCI field

Chapter 3

Machine Learning Algorithms

3.1 Introduction

This chapter investigates the machine learning and signal processing techniques that have been used in the implementation of the algorithms used in this research. Firstly, the classifiers such as Bayesian quadratic classifier, Bayesian network, HMM, neural network and Fisher linear classifier will be explained.

In the last section the signal processing techniques will be evaluated. First, data splitting techniques such as cross validation will be explained. These techniques are especially important for smaller datasets, such as the Purdue EEG dataset, used in this research. Independent Component Analysis (ICA) and feature extraction techniques, which include autoregressive coefficients (AR) and adaptive autoregressive coefficients (AAR) conclude this section.

3.2 Bayesian Classifiers

Given a classification task of N classes, $\omega_1, \omega_2, \cdots \omega_N$ and an unknown pattern, that is represented by feature vector x, the N conditional probabilities $P(\omega_k \mid x)$ is formed

where $k = 1, 2 \cdots N$ and it is called a posteriori probability. Each of them represents the probability that the unknown pattern belongs to the respective class ω_k, given that the feature vector takes the value x.

These classifiers calculate either the maximum of these N values or equivalently, the maximum of an appropriately defined function of them and assigns the unknown pattern to the class corresponding to the maximum. As will follow in the next section the Bayes rule will facilitate this calculation. Before going into the details of this method and as the EEG signal will be modeled by Gaussians, some preliminary equations related to Gaussians will be evaluated.

3.2.1 Gaussian Density and Likelihood Calculation

The Gaussian probability density function (pdf) for the d-dimensional random variable x is shown as $N(\mu, \Sigma)$ and is calculated by:

$$f_{(\mu,\Sigma)}(x) = \frac{1}{\sqrt{2\pi}^d \sqrt{det(\Sigma)}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1} (X-\mu)} \tag{3.1}$$

Where μ is the mean vector and Σ is the covariance matrix and they are parameters of the Gaussian distribution. The mean vector μ contains the mean values of each dimension, $\mu_i = E(x_i)$, and $E(x)$ is the expected value of x. All of the variances c_{ii} and covariances c_{ij} are collected together into the covariance matrix Σ of dimension $d \times d$:

$$\Sigma = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix} \tag{3.2}$$

The covariance c_{ij} of two components x_i and x_j of x measures their tendency to vary

together,

$$c_{ij} = E((x_i - \mu_i)^T(x_j - \mu_j)).$$

If two components x_i and x_j have zero covariance $c_{ij} = 0$ they are considered as orthogonal. If all components of x are mutually orthogonal the covariance matrix will be diagonal.

The likelihood of a sample point x_i given a data model such as set of parameters Θ for the model pdf, is the value of the pdf $p(x_i|\Theta)$ for that point. In the case of Gaussian models $\Theta = (\mu, \Sigma)$, this is equal to evaluation of equation 3-1.

Joint likelihood: for a set of independent samples $X = \{x_1, x_2, \ldots, x_N\}$, the joint likelihood is the product of the likelihoods for each individual point. For example, in the Gaussian case:

$$p(X \mid \Theta) = \prod_{i=1}^{N} p(x_i|\Theta) = \prod_{i=1}^{N} p(x_i|\mu, \Sigma) = \prod_{i=1}^{N} f_{(\mu,\Sigma)}(x_i) \tag{3.3}$$

If the above equation is differentiated with regard to Θ parameters, μ and Σ, the following Maximum Likelihood estimates for mean and covariance will be calculated:

- ML Mean estimator: $\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i$
- ML Covariance estimator: $\hat{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu)^T(x_i - \mu)$

Computing the log of the likelihood equation above turns the product into a sum:

$$p(X|\Theta) = \prod_{i=1}^{N} p(x_i|\Theta) \quad \Leftrightarrow \quad \log p(X|\Theta) = \log \prod_{i=1}^{N} p(x_i|\Theta) = \sum_{i=1}^{N} \log p(x_i|\Theta)$$

when the pdf is Gaussian, it also avoids the computation of the exponential:

$$\begin{aligned} p(x|\Theta) &= \frac{1}{\sqrt{2\pi}^d \sqrt{\det(\Sigma)}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \\ \log p(x|\Theta) &= \frac{1}{2}\left[-d\log(2\pi) - \log(\det(\Sigma)) - (x-\mu)^T\Sigma^{-1}(x-\mu)\right] \end{aligned} \quad (3.4)$$

since $\log(x)$ is a monotonically growing function, the log-likelihoods have the same relations of order as the likelihoods do:

$$p(x|\Theta_1) > p(x|\Theta_2) \quad \Leftrightarrow \quad \log p(x|\Theta_1) > \log p(x|\Theta_2),$$

So, they can be used directly for classification purposes as was done for EEG mental tasks as well.

3.2.2 Bayes Law

Now the Bayes decision rule is used to classify x_i from a data sample (or several feature vectors X) as belonging to a certain class w_k. This is called Bayesian quadratic classifier and the following equations can be used to calculate it:

$$X \in w_k \quad \text{if} \quad P(w_k|X,\Theta) \geq P(w_j|X,\Theta), \quad \forall j \neq k$$

As stated before, given a set of classes w_k, characterized by a set of known parameters in model Θ, feature vectors X belongs to the class which has the maximum probability once it is known that the sample X is observed. As was seen before $P(w_k|X,\Theta)$ is called the a posteriori probability, because it depends on having seen the observations, as opposed to the a priori probability $P(w_k|\Theta)$, which does not depend on any observation and depends on knowing how to characterize all the classes w_k that means knowing the parameter set Θ.

For most practical classification tasks (e.g. EEG classification), it is practical to make use of likelihoods rather than trying to directly estimate the posterior probability $P(w_k|X,\Theta)$. According to Bayes law:

$$P(w_k|X,\Theta) = \frac{p(X|w_k,\Theta)\ P(w_k|\Theta)}{p(X|\Theta)} \tag{3.5}$$

where w_k is a class, X is a sample containing one or more feature vectors and Θ is the parameter set of all the class models. Since the denominator is independent of ω_k, so the same for all classes they can be dropped from the equation and know that $P(w_k|X,\Theta)$ is proportionate to:

$$P(w_k|X,\Theta) \propto p(X|w_k,\Theta)\ P(w_k|\Theta), \quad \forall k$$

or if take log from both side:

$$\log P(w_k|X,\Theta) \propto \log p(X|w_k,\Theta) + \log P(w_k|\Theta) \tag{3.6}$$

3.2.3 Mixtures of Gaussians

Gaussian Mixtures are combinations of Gaussian distributions. A mixture of Gaussians can be written as a weighted sum of Gaussian densities. A weighted mixture of K Gaussians can be written as

$$fm(x) = \sum_{k=1}^{K} \pi_k \cdot f_{(\mu_k,\Sigma_k)}(x), \tag{3.7}$$

where the weights are all positive and sum to one:

$$\pi_k \geq 0 \quad \text{and} \quad \sum_{k=1}^{K} \pi_k = 1 \qquad \text{for} \quad k \in \{1,\ldots,K\}. \tag{3.8}$$

By varying the number of Gaussians K, the weights π_k, and the parameters μ_k and Σ_k of each Gaussian density function, Gaussian mixtures can be used to describe any complex probability density function. The process of changing these parameters is training of the Gaussian mixture.

To find these parameters to optimally fit a certain probability density function for a set of data, an iterative algorithm like EM can be used. Starting with initial values for all parameters they are re-estimated iteratively using EM. The mixture of Gaussians has been used in conjunction with Bayesian network and HMM as will be explained in chapter four.

3.3 Bayesian Networks

Bayesian Network is a modeling tool that combines directed acyclic graphs with Bayesian probability. A directed graph is acyclic if there is no directed path $A_1 \rightarrow \ldots \rightarrow A_n \quad \textit{while} \ A_1 = A_n$. The Bayesian network consists of the followings [49]:

- A set of variables and a set of directed edges between variables.
- Each variable has a finite set of mutually exclusive states.
- The variables together with the directed edges form a directed acyclic graph.
- To each variable A with parents $B_1 \ldots, B_n$ there is attached the potential table $P(A|B_1, \ldots, B_n)$

Figure 3.1 shows a simple example of Bayesian network which consists of a causal graph combined with an underlying probability distribution. Each node of the network in the figure corresponds to a variable and edge represents causality between these events.

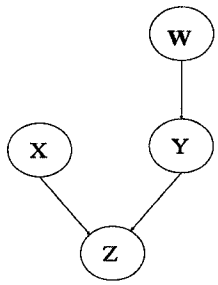

Figure 3.1: Bayesian Networks

The other element of a Bayesian network is the probability distributions associated with each node. With this information the network can model probabilities of complex causal relationships. In Figure 3.1 there are set of states (Z,X,Y,W) with arrows representing conditional dependence between them. in general according to chain rule:

$$P(Z,X,Y,W) = P(Z \mid Y,X,W)P(Y \mid W,X)P(X \mid W)P(W), \tag{3.9}$$

However, for the above Bayesian network considering the causal relationships, this simplifies to:

$$P(Z,X,Y,W) = P(Z|Y,X)P(Y|W)P(X)P(W), \tag{3.10}$$

From a mathematical point of view, the basic property of Bayesian network is the chain rule as mentioned above: a Bayesian network is a compact representation of the joint probability table over its universe. From an engineering point of view, a Bayesian network is a type of graphical model. The structure of the network is formulated in a graphical communication language for which the language features have very simple semantics known as causality.

Furthermore, the graphical specification also specifies the requirements for the quantitative part of the model. Specifying the structure of a Bayesian network consists of two parts: specifying the network topology and estimating the parameters of the conditional probability density function. These parameter estimations can be

performed by the EM algorithm.

A Bayesian network can be constructed to model the probability density function of each class $p(x \mid w_k)$ separately. These densities may be substituted into Bayes rule to obtain estimates of the posterior probabilities of class membership. This estimation is done for EEG signal, which will be explained in the next chapter.

Bayesian networks provide a graphical representation of the variables in a problem and the relationship among them. This representation needs to be specified or learned from data. This structure together with the conditional density functions, allows the multivariate density function to be specified through the product rule in equation 3.10.

3.4 Hidden Markov Models

HMM is a statistical model in which the system being modeled is assumed to be a Markov process with unknown parameters, and the goal is to determine the hidden parameters, from the observable parameters. HMM has applications in speech recognition, image processing and pattern recognition.

The HMM is a finite set of states, each of which is associated with a probability distribution. In this research, this probability distribution is multi-dimensional. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution.

HMM Model is specified by the set of states $S = \{s_1, s_2, \ldots, s_{N_s}\}$, and a set of parameters $\Theta = \{\pi, \mathbf{A}, \mathbf{B}\}$:

- The prior probabilities $\pi_i = P(q_1 = s_i)$ are the probabilities of s_i being the first state of a state sequence. They are collected in the vector π .

- The transition probabilities are the probabilities to go from state i to state j: $a_{i,j} = P(q_{n+1} = s_j | q_n = s_i)$. They are collected in the matrix $\mathbf{A}$.

- The emission probabilities characterize the likelihood of a certain observation x, if the model is in state s_i. Depending on the kind of observation x:

  - for discrete observations, $x_n \in \{o_1, \ldots, o_K\}$.$b_{i,k} = P(x_n = o_k | q_n = s_i)$, the probabilities to observe v_k if the current state is $q_n = s_i$. The numbers $b_{i,k}$ can be collected in a matrix $\mathbf{B}$.

  - for continuous valued observations, e.g., $x_n \in \mathbb{R}^D$: A set of functions $b_i(x_n) = p(x_n | q_n = s_i)$ describing the probability densities over the observation space for the system being in state s_i. They are collected in the vector $\mathbf{B}(x)$ of functions. Emission pdfs are often parameterized. In this work, this parameterization is done by mixtures of Gaussians as explained in section 3.2.3.

The operation of a HMM is characterized by:

- The hidden state sequence $Q = \{q_1, q_2, \ldots, q_N\}, \quad q_n \in S$.

- The observation sequence $X = \{x_1, x_2, \ldots, x_N\}$.

In this work an HMM model is assigned to each of the mental tasks that will be explained in the next chapter. Finally the likelihood of test vectors are calculated with respect to these models. The test vector is assigned to a mental task with greater value of liklihood [47].

3.5 Fisher Linear Discriminant Analysis

Linear Discriminant Analysis is a method of classification that uses a weighted sum. For each object that is to be classified, linear discriminant analysis takes a weighted

sum of values of the variables that determine the classification. The value of the weighted sum is then used to determine the classification results. The Fisher method is one of the methods for calculation of this weighted sum and will be explained in this section.

Supposeing that there is a set of training patterns $x_1, x_2 ... x_N$, each of which is assigned to one of two classes, w_1 or w_2 then using this design set, a weight vector $\mathbf{w}$ and a threshold w_0 is defined such that

$$\mathbf{w}^T\mathbf{x} + w_0 \begin{cases} > 0 \Rightarrow \mathbf{x} \in w_1 \\ < 0 \Rightarrow \mathbf{x} \in w_2 \end{cases} \tag{3.11}$$

if $\mathbf{z} = (1, x_1, ..., x_N)$ and $\mathbf{v} = (w_0, w_1, ..., w_p)$:

$$\mathbf{v}^T\mathbf{z} \begin{cases} > 0 \Rightarrow \mathbf{x} \in w_1 \\ < 0 \Rightarrow \mathbf{x} \in w_2 \end{cases} \tag{3.12}$$

A sample in class w_1 is classified correctly if $\mathbf{v}^T\mathbf{z} > 0$ if all values in w_2 are replaced with their opposite sign values $\mathbf{t}$:

$$\mathbf{v}^T\mathbf{t} > 0 \qquad \mathbf{t}_i^T = (1, \mathbf{x}_i^T), \mathbf{x}_i \in w_1 | \mathbf{t}_i^T = (-1, -\mathbf{x}_i^T), \mathbf{x}_i \in w_2 \tag{3.13}$$

So, a classification is found that makes $\mathbf{v}^T\mathbf{t}$ positive for as many samples in the design set as possible. The simplest criterion to minimize is the perceptron criterion function

$$\mathbf{J}_P(\mathbf{v}) = \sum_{t_i \in T} -\mathbf{v}^T\mathbf{t}_i \tag{3.14}$$

where $T = \{\mathbf{t}_i | \mathbf{v}^T\mathbf{t}_i < 0\}$ so J_P (considering the dot product concept) is proportional to the sum of the distances of the misclassified samples to the decision boundary.

The approach taken by Fisher was to find a linear combination of the variables that separates the two classes as much as possible. That is, a direction is found along which the two classes are best separated in some sense. The criterion proposed by Fisher is the ratio between-class to with-in class variances. Formally, a direction w is found such that

$$J_F = \frac{|\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)|^2}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} \tag{3.15}$$

is maximized, where m_1 and m_2 are the group means and S_w is the pooled within-class sample covariance matrix

$$S_W = \Sigma_1 + \Sigma_2 \tag{3.16}$$

Σ_1 and Σ_2 are the maximum likelihood estimates of covariance matrixes of classes w_1 and w_2 respectively. Maximizing the above criterion gives a solution for the direction w. The solution for w that maximizes J_F can be obtained by differentiating J_F with respect to w and equating it to zero, which gives us this:

$$\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{3.17}$$

To implement the Fisher linear discriminant classifier, the statistical pattern recognition toolbox of Matlab was used [50].

3.6 Neural Networks

In this section the basic concept of neural network is explained. A feedforward neural network, trained by error back propagation, was used in this works to classify mental tasks. So, at the end of this section there is a brief introduction to error backprop-

agation algorithm. The more mathematical treatment of backpropagation algorithm can be found in [51].

3.6.1 Basics of Neural Networks

A neural network is a massively parallel distributed computer that is made of simple processing units called neurons. It resembles the brain in two ways. The knowledge is acquired from the experiment through a learning process and the knowledge is stored in inter-neuron connections or synaptic weights.

For directed graphs, a recurrent architecture can be distinguished (containing cycles) and feedforward architectures, which is acyclic. A very important special case of feedforward networks is given by layered networks, in which the nodes of the graph are organized into an ordered series of disjoint classes (the layers) such that connections are possible only between elements of two consecutive classes and following the natural order.

The weight between the unit k and the unit j of a network is indicated with w_{kj}, and it is assumed that all elements of a layer are connected to all elements of the successive layer. In this way, the connections between two layers can be represented by a weight matrix W. In this matrix the entry j_k corresponds to the connection between node j and node k in successive layer.

In a layered network, the function is computed sequentially, assigning the value of the argument to the input layer, and then calculating the activation level of the successive layers as will be described next, until the output layer is reached. The output of the function computed by the network is the activation value of the output units.

All units in a layer are updated simultaneously, and all the layers are updated sequentially, based on the state of the previous layer. Each unit k calculates its

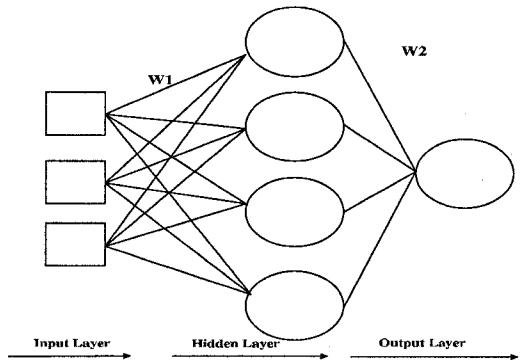

Figure 3.2: Feedforward Neural Network

value y_k by a linear combination of the values at the previous layer, x, followed by a nonlinear transformation as follows: $y_k = \phi(Wx)$, where ϕ is called the transfer function and for which a common choice is the logistic function

$$f(n) = \frac{1}{1 + e^{-n}}$$

Notice that the input/output behavior of the network is determined by the weights and training the network amounts to automatically, choosing the values of the weights.

Given a training set of data and a fixed error function for the performances of the network, the training of a neural network can be done by finding those weights that minimize the network's error on such a sample. This training can be done by gradient descent, if the error function is differentiable, by means of the backpropagation algorithm that is explained in the next section.

3.6.2 Back Propagation Algorithm

Basically, the error backpropagation algorithm consists of two passes. These two passes are forward and backward passes. In the forward pass an input vector is applied to the input neurons and its effect propagates through the network layer by layer. At the end, a set of outputs is produced as the actual response of the neural

network [51].

During the forward pass the synaptic weights of the networks are not changed. During the backward pass, on the other hand, the synaptic weights are all adjusted in accordance with an error-correction rule. Specifically, the actual response of the network is subtracted from a desired response to produce an error signal. This error signal is then propagated backward through the network, hence the name backpropagation.

Backpropagation provides a way to compute the necessary gradients, so that the network finds a local minimum of the training error function with respect to network weights. The chain rule of differentiation is used to compute the gradient of the error function with respect to the weights.

If y_i is the value of the ith unit, for each w_{ij} connecting it to the previous layer's units, one can write the partial derivative of the error function as

$$\frac{\delta E}{\delta w_{ij}} = \frac{\delta E}{\delta y_i}\frac{\delta y_i}{\delta w_{ij}} = \frac{\delta E}{\delta y_i}\phi'(x_i)x_i = \epsilon_i$$

where ϕ is the tansfer function defined before, and hence the update for such weight will be $\Delta w_{ij} = -\eta\epsilon_i y_i$ where η is a parameter known as the learning rate. A more detailed mathematical backgrounds of error backpropagation algorithm is presented in [51].

Among the main problems of neural networks is that, the training algorithm is only guaranteed to converge to a local minimum and the solution is affected by the initial conditions.

Another problem is the design of the architecture, which often is chosen as the result of trial and error. The problem of neural network architecture was previously evaluated in [4]. It was found that for the Purdue dataset a two-layer feedforward neural network with 20 neurons in the hidden layer is the best architecture. So, in

this research the same architecture is used [51].

3.7 Signal Processing Algorithms

In this section, a set of general signal processing techniques for pre-processing, feature extraction and testing of the classifiers, are explained.

3.7.1 Data Splitting Techniques

Using flexible models, such as neural networks, it is widely accepted that in many cases there is not sufficient data available to reliably generalize the results. In this thesis, this lack of data can be especially seen in the Purdue EEG dataset. This gives rise to the problem that how data should be split to maximize generalization abilities.

In this work the dataset D is split into two parts, the training T on which the model parameters are trained and the test sets H for final assessment of model which has been designed on T. Given a data set $D = \{x(i); t(i)\}_{i=1}^{N}$ of N independent input-output examples, there are four methods to split the data into the test, H, and training sets T.

- **Hold-Out Cross Validation** The data is split at once into T and H. If α is the splitting ratio then $N_F = \alpha N$ for testing and $N_T = (1 - \alpha)N$ for training.

- **K-fold cross validation** In this method, the dataset is split into K disjoint subsets F_j of approximately equal size, $\bigcup_{j=1}^{K} F_j = D$. For $\alpha < 0.5$, the split ratio is the size of the subset to the total amount of data, ie, $K = [1/\alpha]$. On each subset the model is evaluated and designed on the remaining data.

- **Leave One Out (LOO)** This method is a special case of the previous method when $\alpha = 1/N$, meaning that one point of the set is put in the test set and

training is done on the rest of the data set and in later repetitions this point is changed with other points in the dataset. This method has been used in few cases in this research.

- **Randomized permutation cross validation** This involves resampling test sets by randomly selecting $N_F = N\alpha$ samples for the test set, and the rest for the design set. This can be repeated at most $K \leq \binom{N}{N_F}$ times.

For different problems considering the generalization and model consistency different choices of α might lead to optimal solutions. In this work, considering the related research [15] a 5-fold cross validation is implemented meaning that 1/5 of the data was considered for testing and 4/5 for training.

To conform with [15] the data was split into 10 disjoint subsets. Two of these subsets were considered for the test set and 8 remaining for the training sets. By considering all combinations of these 10 subsets $\binom{10}{2} = 45$, all the algorithms were repeated 44 times giving almost an equal chance for each of these subsets to appear in the test and training sets.

For finding the best neural network architecture, previously the LOO method was used but this is very time consuming considering the number of dataset points in all current experiments. So, LOO was substituted by the 5-fold cross validation as explained earlier.

3.7.2 Feature Extraction Methods

In this section the feature extraction methods will be explained. In past research many different feature extraction methods [4] were used but in the present research only AR and adaptive AR coefficients are used. AR coefficients were used for Purdue dataset and AAR is used for the Graz dataset.

Autoregressive Coefficients

A large class of discrete processes can be modeled in the form of

$$y_n = \sum_{i=1}^{N} a_i y_{n-1} + \xi_n \tag{3.18}$$

where y_n are samples of the process that are going to be modeled, and ξ_n is a white noise sequence, with the autocorrelation function as:

$$\mathbf{R}_{\epsilon\epsilon} = \begin{cases} 0 & otherwise \\ \sigma_\epsilon^2 & k = 0 \end{cases} \tag{3.19}$$

the above model is called an Nth-order autoregressive model and is denoted by AR(N). This model can be used to describe the EEG as filtered white noise [41].

The autocorrelation of any process can tell a lot about the sample to sample behavior of a sequence. In case of white noise by above autocorrelation function there is essentially no sample-to-sample correlation. The autocorrelation function for the AR(N) process can be obtained as follows:

$$R_{yy}(k) = E[y_n y_{n-k}] \tag{3.20}$$

$$= E\left[\left(\sum_{i=1}^{N} a_i y_{n-i} + \epsilon_n\right)(y_{n-k})\right] \tag{3.21}$$

$$= E\left[\sum_{i=1}^{N} a_i y_{n-i} y_{n-k}\right] + E[\epsilon_n y_{n-k}] \tag{3.22}$$

$$= \begin{cases} \sum_{i=1}^{N} a_i R_{yy}(k-i) & k > 0 \\ \sum_{i=1}^{N} a_i R_{yy}(i) + \sigma_\epsilon^2 & k = 0 \end{cases} \tag{3.23}$$

If the values of the autocorrelation function are known for $k = 0, 1, 2...N$, a set of equations can be used, resulting from the above equations to calculate AR(N) coefficients. AR coefficients have been used extensively in EEG analysis [15].

Adaptive Autoregressive Coefficients

The AR coefficients explained before are based on the assumption of the EEG being stationary. In reality the EEG is a non-stationary signal meaning that its statistics change over time. To consider this fact, A. Schloegl [42] [43] considered defining time varying AR parameters. To calculate such parameters they used adaptive techniques so these coefficients were called Adaptive Autoregressive coefficients or AAR.

In the present research a recursive method has been used for calculation of AAR coefficients. The details of these methods can be found in [43]. The only difference between AAR and the conventional AR model is that the AAR parameters are allowed to vary in time as can be seen in the following:

$$y_n = \sum_{i=1}^{N} a_i(n) y_n(i) + \xi \tag{3.24}$$

It is assumed that only a small non-stationarity takes place that is called nearly stationary. Some upper limit of adaptation rate can be assumed and this means that AR parameters change only slowly with time.

It is assumed that the changes of the AAR parameters within one iteration are smaller than the estimation error. If the assumption is fulfilled, the process is nearly

stationary then an AAR model can be used to describe the time-variation in the data. If those assumption are not fulfilled, the process is highly non-stationary and some transient event occurs which can not be described by the AAR parameter [42].

The only difference of AAR to the conventional AR model is that the AAR parameters are allowed to vary in time An AAR model with order N is written as

$$y(t) = a_1(t)y(t-1) + ... + a_N(t)y(t-N) + \xi =$$

$$\mathbf{a}^T(t) * \mathbf{Y}(t-1) + \xi \qquad i = 1, 2...N$$

ξ is a zero-mean Gaussian noise process with variance σ_x^2 t corresponds to time. The difference with (stationary) autoregressive (AR) model is that the AAR parameters vary with time. The one-step prediction error is

$$e(t) = y(t) - \mathbf{a}'(t-1)^T * \mathbf{Y}(t-1) \tag{3.25}$$

in practice, the AAR parameters are only estimated values a'_k. The difference between the prediction error e(t) and the innovation process x(t) is that in the former one the estimated parameters rather than the "true" model parameters are used.

Algorithms for Calculating AAR coefficients

AAR parameters are estimated using a variety of adaptive algorithms.
Taking UC to be the update coefficient and $k(t)$ the update gain the following algorithms have been proposed:

Least Mean Square I (LMS I): The LMS algorithm is described in the following

update equation:

$$\mathbf{a}'(t) = \mathbf{a}'(t-1) + UC/MSY * e(t) * \mathbf{Y}(t-1) \tag{3.26}$$

MSY is the variance of the signal y.

Least Mean Square II (LMS II): The adaptation equation 3.26 uses a constant adaptation rate of UC/MSY in LMS II a time-varying adaptation rate is used in a way that also normalization factor is esimated adaptively as in:

$$\mathbf{V}_k = (1 - \mathbf{UC}) * \mathbf{V}_{k-1} + \mathbf{UC} * e_k^2 \tag{3.27}$$

and the update equation is given in Equation 3.28.

$$\mathbf{a}'(t) = \mathbf{a}'(t-1) + UC/\mathbf{V}_k * e(t) * \mathbf{Y}(t-1) \tag{3.28}$$

Recursive AR (RAR): RAR is a recursive method for estimating the time-varying AR parameter as in equations:

$$\mathbf{A}(t) = (1 - UC) * \mathbf{A}(t) + UC * \mathbf{Y}(t) * \mathbf{Y}(t)^T \tag{3.29}$$

$$\mathbf{k}(t) = UC * \mathbf{A}(t) * \mathbf{Y}(t)/(UC * \mathbf{Y}(t)^T * \mathbf{A}(t) * \mathbf{Y}(t) + 1) \tag{3.30}$$

$$\mathbf{a}'(t) = \mathbf{a}'(t-1) + \mathbf{k}(t)^T * e(t) \tag{3.31}$$

The various AAR estimation algorithms differ in how $\mathbf{k}(t)$ is calculated. The mean

squared error (MSE) is used to measure how well the AAR estimates describe the observed process y. Normalizing the MSE by the variance of the signal (MSY), gives a relative error variance REV, being a criterion for the goodness-of-fit. For other AAR techniques such as Kalman filtering you can refer to [42] and [43].

3.7.3 Information Theory Basics

In the past, the performance of EEG based brain computer interfaces were quantified mostly by calculation of the correct classification percentage. It also looks reasonable to consider the BCI as a communication channel and to quantify the information taken from it. In this the information theoretic basis of this quantification is presented.

An event X is a set of outcomes of a random experiment. If $P(X)$ is the probability that the event X will occur, then the self-information corresponding to X is given by:

$$i(X) = -log_b P(X) \tag{3.32}$$

where b is equal to two, the unit of information is called bit. According to the above equation if the probability of an event is low, the amount of information associated with it is high and vice versa, if the probability of an event is high the amount of information assigned to it is low.

For example, consider a house equipped with alarm. If there is a burglary in this house then the event of the alarm ringing does not carry lots of information but the event of the alarm not ringing carries more information as it is less probable to happen.

If there are a set of independent events X_i, which are sets of outcomes of some experiment S, such that $\bigcup X_i = S$ where S is the sample space, then the average

self-information of the random experiment is called entropy and is given by

$$H = \Sigma P(X_i) i(X_i) = -\Sigma P(X_i) log_b P(X_i) \tag{3.33}$$

for the countinuous stochastic x process the sigma is replaced by the integral. It can be shown that the entropy of a stochastic process x with a given variance σ_x is

$$H(x) \leq 0.5 * log_2(2\pi e \sigma_x) \tag{3.34}$$

When x is Gaussian the equality holds. So, this gives the maximum of entropy. Entropy is the basic concept of information theory. The entropy of a random variable can be interpreted as the degree of information that the observation of the variable gives. The more random, unpredictable and unstructured the variable is, the larger its entropy. There is one more quantity called mutual information (MI) which is defined as:

$$i(x_k; y_i) = log \left[\frac{P(x_k|y_i)}{P(x_k)} \right] \tag{3.35}$$

Normally the averaged value of this quantity is used and represented in entropy format as:

$$I(X;Y) = H(X) - H(X|Y) \tag{3.36}$$

The average MI is the entropy of the source minus the uncertainty that remains about the source output after the reconstructed value has been received. The MI is a natural measure of the dependence between random variables. It is always non-negative, and zero if and only if the variables are statistically independent.

It is shown experimentally that MI is superior to other methods for BCI applica-

tions [44]. So, in section 4.3.2 MI is calculated for the Graz EEG dataset.

3.7.4 Expectation Maximization Algorithm

EM is an iterative optimization method to estimate some unknown parameters, Θ given measurement data X. EM is typically used to compute maximum likelihood estimates given incomplete samples. In this thesis as will follow we will use EM algorithm for training the hidden Markov model and Bayesian networks. The EM algorithm consists of two primary steps: an expectation step, followed by a maximization step. The expectation is obtained with respect to the unknown underlying variables, using the current estimate of the parameters and conditioned upon the observations. The maximization step then provides a new estimate of the parameters. These two steps are iterated until convergence. The algorithm works as follows:

1. Start from K initial Gaussian models $\mathcal{N}(\mu_k, \Sigma_k)$, $k = 1 \ldots K$, with equal priors set to $P(w_k) = 1/K$.

2. **Estimation step**: compute the probability $P(w_k^{(i)}|x_n, \Theta^{(i)})$ for each data point x_n to belong to the class $w_k^{(i)}$:

$$P(w_k^{(i)}|x_n, \Theta^{(i)}) = \frac{P(w_k^{(i)}|\Theta^{(i)}) \cdot p(x_n|w_k^{(i)}, \Theta^{(i)})}{p(x_n|\Theta^{(i)})} \tag{3.37}$$

$$= \frac{P(w_k^{(i)}|\Theta^{(i)}) \cdot p(x_n|\mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_j P(w_j^{(i)}|\Theta^{(i)}) \cdot p(x_n|\mu_j^{(i)}, \Sigma_j^{(i)})} \tag{3.38}$$

3. **Maximization step**:

   (a) update the means:

$$\mu_k^{(i+1)} = \frac{\sum_{n=1}^{N} x_n P(w_k^{(i)}|x_n, \Theta^{(i)})}{\sum_{n=1}^{N} P(w_k^{(i)}|x_n, \Theta^{(i)})}$$

(b) update the variances:

$$\Sigma_k^{(i+1)} = \frac{\sum_{n=1}^{N} P(w_k^{(i)}|x_n, \Theta^{(i)})\ (x_n - \mu_k^{(i+1)})(x_n - \mu_k^{(i+1)})}{\sum_{n=1}^{N} P(w_k^{(i)}|x_n, \Theta^{(i)})}$$

(c) update the priors:

$$P(w_k^{(i+1)}|\Theta^{(i+1)}) = \frac{1}{N}\sum_{n=1}^{N} P(w_k^{(i)}|x_n, \Theta^{(i)})$$

In the present case, all the data points participate to the update of all the models, but their participation is weighted by the value of $P(w_k^{(i)}|x_n, \Theta^{(i)})$.

4. Go to step 2.

End: the parameter estimate has converged or total likelihood increase for the training data falls under some preset threshold.

The global criterion in the present case is the joint likelihood of all data with respect to all the models:

$$\begin{aligned}\mathcal{L}(\Theta) = \log p(X|\Theta) &= \log \sum_{Q} p(X, Q|\Theta) \\ &= \log \sum_{Q} P(Q|X, \Theta) p(X|\Theta) \quad \text{(Bayes)} \\ &= \log \sum_{k=1}^{K} P(w_k|X, \Theta) p(X|\Theta)\end{aligned}$$

3.7.5 Independent Component Analysis

ICA is a very general-purpose statistical technique in which observed random data are linearly transformed into components that are maximally independent from each other. ICA can be formulated as the estimation of a latent variable model. The

intuitive notion of maximum non-gaussianity can be used to derive different objective functions whose optimization enables the estimation of the ICA model.

Alternatively, one may use more classical notions like maximum likelihood estimation or minimization of MI to estimate ICA, somewhat surprisingly; these approaches are (approximately) equivalent.

ICA is very closely related to the method called blind source separation. Here, a source means an original signal or the independent component. Blind means that very little is known about the mixing matrix, and make little assumptions on the source signals. ICA is one method, perhaps the most widely used, for performing blind source separation.

Applications of ICA can be found in many different areas such as audio processing, biomedical signal processing, image processing, telecommunications, and econometrics. EEG potentials are presumably generated by mixing some underlying components of brain activity. This situation is quite similar to a typical ICA problem used to find the original components of brain activity, although only mixtures of the components can be observed. For EEG one of these components can be the eye artifacts which are separated in this and other studies, using ICA technique [34].

In this section the mathematical backgroud of ICA is explained but a more detailed treatment of the problem can be found in [7] and [45].

ICA Definition

Let us denote by $\mathbf{y}$ the random vector whose elements are the mixtures $y_1, ..., y_N$, and likewise by $\mathbf{s}$ the random vector with elements $s_1, ..., s_N$. Let us denote by $\mathbf{A}$ the matrix with elements a_{ij} . Using this vector-matrix notation:

$$\mathbf{y} = \mathbf{A}\mathbf{s} \tag{3.39}$$

So, The observed values $y_j(t)$ are then a sample of this random vector as in:

$$y_j = a_{j1}s_1 + a_{j2}s_2 + ... + a_{jn}s_N, \qquad forall j \tag{3.40}$$

The model in Equation 3.39 and 3.40 is called independent component analysis, or ICA. The ICA describes how the observed data are generated by a process of mixing the components s_i so, it is a generative model.

The independent components are latent variables, meaning that they cannot be directly observed. Also the mixing matrix is assumed to be unknown. The random vector $\mathbf{y}$, is the only thing that can be observed and both $\mathbf{A}$ and $\mathbf{s}$ must be estimated using this observed vector.

The starting point for ICA is the simple assumption that the components s_i are statistically independent. After estimating the matrix A, its inverse can be computed, W, and the independent components obtained by:

$$\mathbf{s} = \mathbf{W}\mathbf{y} \tag{3.41}$$

3.8 ICA Estimation

In this section ICA estimation methods are briefly introduced.

3.8.1 ICA Estimation Criterion

To estimate $\mathbf{A}$ in Equation 3.39 there is an essential restriction in ICA that the independent components must be non-gaussian. This is the key issue in ICA model estimation.

The central limit theorem [52], says that the distribution of a sum of independent random variables tends toward a gaussian distribution. So, a sum of two independent

random variables usually has a distribution that is closer to a gaussian distribution than any of the two original random variables.

To estimate one of the independent components, a linear combination is considered of the y_i as in Equation 3.41. This is denoted by $x = \mathbf{w}^T\mathbf{y} = \sum_i w_i y_i$. w is a vector to be estimated. If $\mathbf{w}$ were one of the rows of the inverse of $\mathbf{A}$, this linear combination would actually equal one of the independent components.

The ideal goal here is to use the central limit theorem to determine $\mathbf{w}$ so that it would equal one of the rows of the inverse of $\mathbf{A}$. Practically, such a w can not be determined exactly, because there is no knowledge of matrix A, but an estimator can be found that gives a good approximation of it.

This leads to the basic principle of ICA estimation. If a change of variables is made, defining $\mathbf{z} = \mathbf{A}^T\mathbf{w}$. Then, $x = \mathbf{w}^T\mathbf{y} = \mathbf{w}^T\mathbf{A}\mathbf{s} = \mathbf{z}^T\mathbf{s}$. So, x is a linear combination of s_i, with weights given by z_i.

Since a sum of even two independent random variables is more gaussian than the original variables, $z^T s$ is more gaussian than any of the s_i and becomes least gaussian when it in fact equals to one of the s_i. In this case, obviously only one of the elements z_i of $\mathbf{z}$ is nonzero. Therefore, $\mathbf{w}$ can be taken as a vector that maximizes the nongaussianity of $w^T y$.

Such a vector would necessarily correspond to a $\mathbf{z}$ which has only one nonzero component. This means that $\mathbf{w}^T\mathbf{y} = \mathbf{z}^T\mathbf{s}$ equals one of the independent components.

So, Maximizing the nongaussianity of $\mathbf{w}^T\mathbf{y}$ gives one of the independent components.

3.8.2 ICA Estimation Methods

To use non-gaussianity in ICA estimation, there must be a quantitative measure of nongaussianity of a random variable. Different ICA techniques are distinct in the way

that non-gaussianity is defined for them [45]:

- **Kurtosis:** The classical measure of non-gaussianity is kurtosis or the fourth-order cumulant. The kurtosis of x is classically defined as $kurt(x) = E\{x^4\} - 3(E\{x^2\})^2$. For a gaussian x, the fourth moment equals $3(E\{x^2\})^2$. Thus, kurtosis is zero for a gaussian random variable. Kurtosis can be both positive or negative. Random variables that have a negative kurtosis are called subgaussian, and those with positive kurtosis are called supergaussian.

- **Negentropy:** Another important measure of nongaussianity is given by negentropy. Negentropy is based on the information theoretic quantity of entropy explained in section 3.7.3. A fundamental result from information theory is that a gaussian variable has the largest entropy among all random variables of equal variance. This means that entropy could be used as a measure of nongaussianity. To obtain a measure of nongaussianity that is zero for a gaussian variable and always nonnegative, a modified version of the definition of entropy called negentropy is defined: $H(\mathbf{x}gauss) - H(\mathbf{x})$ where $\mathbf{x}gauss$ is a Gaussian random variable of the same covariance matrix as $\mathbf{x}$. Negentropy is always non-negative, and it is zero if and only if x has a Gaussian distribution.

- **Mutual Information** The concept of mutual information was explained in section 3.7.3. Since mutual information is the natural information theoretic measure of random variables independence, it can be used as the criterion for finding the ICA transform. In this approach the ICA of a random vector y is defined as an invertible transformation as in Equation 3.41, where the matrix $\mathbf{W}$ is determined so that the mutual information of the transformed components s_i is minimized. It can be shown that finding an invertible transformation $\mathbf{W}$ that minimizes the mutual information is equivalent to finding directions in which the negentropy is maximized.

In this research a Matlab toolbox [46] is used that applies infomax method as ICA estimation technique. Infomax method is closely related to the minimization of mutual information explained earlier [45].

Chapter 4

Results

4.1 Introduction

In this research the classifiers explained in chapter three are applied to two EEG datasets. In this chapter the results of these experiments will be presented and explained. In section 4.2 the results on the Purdue dataset will be presented while in section 4.3, the results on the Graz dataset will be discussed. At the beginning of each section the datasets are explained.

In section 4.2 different combinations of mental tasks for each subject in the Purdue dataset are compared and the best mental task for each subject is determined. At the end of this section the results of application of ICA as the preprocessing block, will be presented.

In Figure 4.5 the general block diagram of algorithms used for the Purdue dataset are explained. In cases where ICA is not used a time filter is used.

In section 4.3 the results on the Graz dataset will be compared to results of other groups who worked on the same data in the BCI2003 competition[1]. To do this comparison, in addition to using the classical Correct Classification Accuracy,

[1]to get more information on this competition and participating groups refer to [3]

the concept of MI is used as well. To calculate Correct Classification Accuracy, the number of correctly classified vectors is divided by the total number of test vectors.

4.2 Purdue Dataset Results

The Purdue dataset is a known EEG dataset used as a reference for several other research works [2][15]. For this dataset the subjects were seated in a sound controlled booth with dim lighting and noiseless fans for ventilation. An elastic electrode cap was used to record signal from positions C3, C4, P3, P4, O1, and O2, defined by the 10-20 system of electrode placement which can be seen in Figure 4.1.

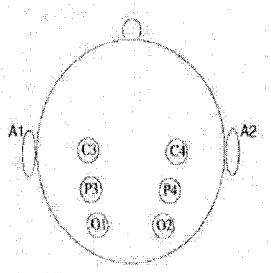

Figure 4.1: The electrode arrangement for the Purdue dataset

The electrodes were connected through a bank of amplifiers. The signal was bandpass filtered from 0.1 to 100 Hz. Data was recorded at a sampling rate of 250 Hz with a 12 bit A/D converter mounted in an IBM-AT computer. Eye blinks were detected and recorded by means of a separate channel of data recorded from two electrodes placed above and below the subject's left eye [35].

The mental tasks were chosen by Keirn and Aunon to invoke hemispheric brain-wave asymmetry [2] and included the following tasks.

Baseline Measurement(B): The subjects were not asked to perform any specific mental task, but to relax as much as possible and think of nothing in particular.

Mental Multiplication(M): The subjects were given nontrivial multiplication

problems and were asked to solve them without vocalizing or making any other physical movements. An example is 49×78. The problems were not repeated and were designed in such a way that an immediate answer was not attainable.

Letter writing(L): The subjects were instructed to mentally compose a letter to a friend or a relative without vocalizing.

Visual counting task(C): The subjects were asked to imagine a blackboard and to visualize numbers being written on it sequentially.

Geometric figure rotation(R): The subjects were asked to visualize a particular three-dimensional block figure being rotated around an axis.

Data were recorded for 10 seconds during each task and each task was repeated five times per session. In Figure 4.2 a sample of the EEG signal during different mental tasks can be seen. Most subjects attended two such sessions recorded on separate weeks, resulting in a total of 10 trials for each task. One of the subjects completed three sessions of data acquisition. This dataset is available online on the internet [35].

4.2.1 Preprocessing with Time Filter

As explained in the previous section for the Purdue dataset a channel was added to record eye blinks by placing an electrode on the forehead above the left browline and another on the left cheekbone. For most of the work on the Purdue dataset this channel was used to remove eye blinks.

As can be seen in Figure 4.3, firstly, the average of the signal over the whole EEG window is calculated and is called $M1$ then the average of the signal over the smaller window of 20 ms is calculated and is called $M2$. The algorithm for removing the affected windows is as follows:

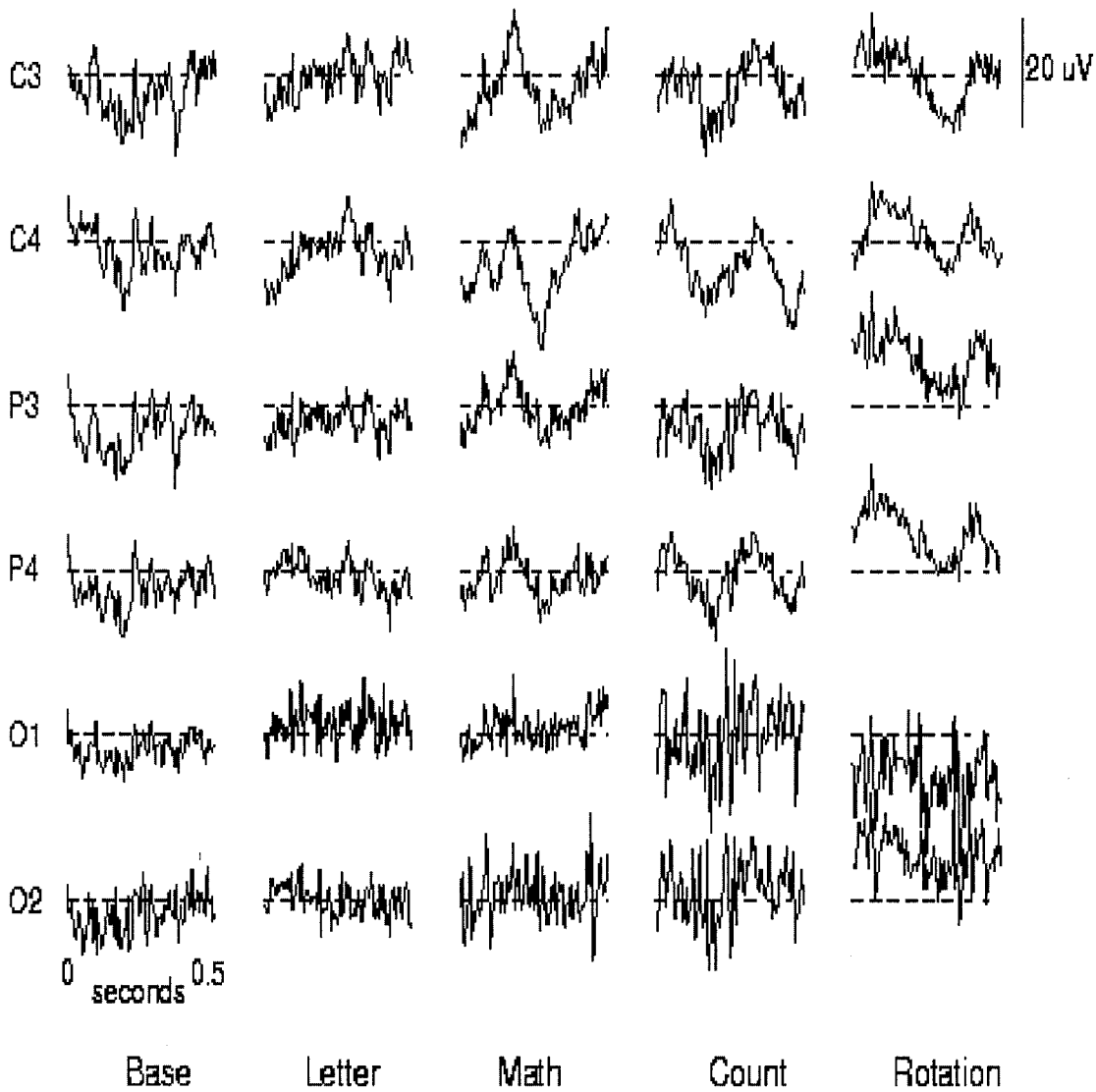

Figure 4.2: Sample of EEG signal in different mental tasks from Purdue dataset [35]

$$\begin{cases} M2 < 2M1 \Rightarrow No - blink \\ M2 > 2M1 \Rightarrow Remove \end{cases}$$

meaning that whenever $M2 > 2M1$ the corresponding window will be removed from the dataset. This can be clearly seen in Figure 4.3.

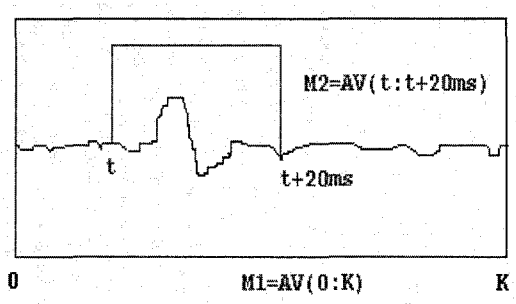

Figure 4.3: Time filter for artifact removal [4]

In Figure 4.5 there is a block diagram of the algorithms used for processing the Purdue EEG dataset. In sections 4.2.2-4.2.6 the time filter, explained above, has been used as the pre-processing stage so in the Figure 4.5 ICA is replaced by this time filter and the rest of the block diagram is the same.

4.2.2 Bayesian Quadratic Classifier

In this part of the research, the Bayesian quadratic classifier has been implemented to classify mental tasks. Given a classification task of N mental tasks, $M_1, M_2, ...M_N$ and an unknown pattern, which is represented by feature vector x, the N conditional probabilities $P(M_k \mid x)$ are calculated where $k = 1, 2 \cdots N$ and P is called a posteriori probability. Each represents the probability that the unknown pattern belongs to the respective mental task M_k, given that the feature vector takes the value x. This value is a vector of numbers which is AR coefficients for the Purdue dataset.

For binary classification of mental tasks $N = 2$, for ternary classification $N = 3$, for quaternary classification $N = 4$. For the Purude dataset M_1, M_2, M_3, M_4, M_5 are B, M, R, C and L respectively.

The classifier calculates either the maximum of these N values or equivalently, the maximum of an appropriately defined function of them (such as logarithm) and assigns

the unknown pattern to the class corresponding to the maximum. As explained in section 3.2, the Bayes law of statistics can be used to calculate the liklihood function:

$$\log P(M_k|X,\Theta) \propto \log p(X|M_k,\Theta) + \log P(M_k|\Theta) \tag{4.1}$$

In this implementation, Θ represents the set of all the means and variances, of the EEG signal in each of mental tasks.

The $p(X|M_k,\Theta)$ and $\log p(X|M_k,\Theta)$ are the joint likelihood and joint log-likelihood of the sample X with respect to the model Θ for class M_k. The probability $P(M_k|\Theta)$ is the a-priori class probability for the class M_k that defines an absolute probability of occurrence for the class M_k.

For calculating the above parameters, the mean and covariance of EEG data was calculated for each of the mental tasks and considered as model parameters. This is equivalent to training the model for neural network or other supervised classifiers.

Later the likelihoods of the test vectors were calculated based on these models and parameters. The vectors were assigned to the model with the greater log-likelihood. More details about this classifier can be found in section 3.2. The results of Bayesian quadratic classifier on the Purdue dataset are represented in Tables 4.1-4.3. In these tables B,M,C,R and L refer to Baseline, Multiplication, visual Counting, Rotation and Letter writing mental tasks respectively, as explained earlier in the Purdue dataset description. For Tables 4.1-4.3, 1,3,5 and 6 refer to subjects 1,3,4 and 6 respectively.

Table 4.1 shows the results for the binary classification of mental tasks (i.e. B,M,C,R and L). In this table B/M refers to the classification of the Baseline versus Multiplication mental task. For subject one this is equal to 93.65 ± 2.19, which is also true for other binary combinations such as B/L, B/C and so on.

Note that for every number pair in each table entry such as 93.6 ± 2.19 the first number (93.65) represents the average, while the second number (2.19) is the corre-

Subjects	1	3	5	6
B/M	93.65 ± 2.19	90.40 ± 3.07	85.49 ± 3.42	97.23 ± 2.27
B/C	98.83 ± 1.04	92.46 ± 3.03	79.81 ± 4.30	89.54 ± 4.55
B/L	92.30 ± 3.12	90.69 ± 2.93	83.95 ± 4.24	83.12 ± 4.93
B/R	96.02 ± 2.93	87.22 ± 4.22	89.06 ± 3.01	96.22 ± 2.42
M/C	96.78 ± 1.98	86.43 ± 4.00	87.28 ± 3.43	98.83 ± 1.55
M/L	96.43 ± 2.48	91.70 ± 3.99	85.94 ± 3.28	94.32 ± 2.48
M/R	98.17 ± 1.61	86.90 ± 3.61	88.55 ± 2.55	93.47 ± 3.20
C/R	85.61 ± 5.39	86.17 ± 4.71	88.07 ± 3.87	94.47 ± 2.08
C/L	88.86 ± 4.72	94.44 ± 2.64	86.97 ± 3.52	85.73 ± 5.02
L/R	91.13 ± 3.28	85.80 ± 3.68	90.66 ± 2.3	91.95 ± 3.76
Mean	93.78 ± 2.87	89.22 ± 3.56	86.58 ± 3.40	92.49 ± 3.23

Table 4.1: Binary mental task classification by Bayesian quadratic classifier

sponding standard deviation of 44 classifications according to our 5-fold cross validation algorithm explained in section 3.7.1.

For each individual classification generated by the 5-fold cross validation algorithm, 1/5 of the data is considered for testing and the remaining 4/5 for training. The data was split into ten disjoint subsets. Two of these subsets were considered as the test sets and eight remaining were considered as the training sets.

All combinations of these ten subsets equals 45, as in $\binom{10}{2} = 45$. The algorithms were performed 44 times for 44 different combinations noted above. Doing this, gives almost an equal chance for each of these subsets to appear in the test and training sets. More details can be found in section 3.7.1.

Table 4.2 shows the results for ternary combinations. In this table the B/M/R ternary combination refers to the classification of Baseline, Multiplication and Rotation tasks together. The same is true for other ternary combinations such as L/M/C and C/M/R and so on. As explained before, each entry in this table also shows the average and standard deviation of 44 classifications according to the 5-fold cross validation algorithm.

Subjects	1	3	5	6
L/R/C	81.73 ± 5.34	81.18 ± 4.57	80.44 ± 3.67	84.29 ± 4.83
L/M/R	90.99 ± 3.55	80.11 ± 4.44	**80.88** ± 2.83	88.39 ± 3.84
C/M/R	88.91 ± 3.95	84.55 ± 5.06	77.64 ± 4.39	87.07 ± 4.68
C/M/l	87.71 ± 4.57	79.63 ± 4.44	80.17 ± 3.40	92.06 ± 3.35
B/L/R	87.90 ± 4.47	81.29 ± 3.80	80.10 ± 3.97	*82.71* ± 4.94
B/R/C	87.54 ± 4.48	81.50 ± 4.65	77.05 ± 3.79	88.83 ± 4.08
B/L/C	88.19 ± 4.10	87.94 ± 3.54	72.20 ± 4.56	*76.67* ± 6.02
B/L/M	89.54 ± 3.34	85.41 ± 3.56	75.43 ± 4.59	84.62 ± 4.16
B/M/R	92.53 ± 2.97	**79.59** ± 4.58	79.96 ± 3.73	92.00 ± 3.23
B/M/C	93.43 ± 2.23	83.14 ± 3.58	73.64 ± 4.14	91.35 ± 3.90
Mean	88.85 ± 3.90	82.44 ± 4.22	77.75 ± 3.91	86.80 ± 4.30

Table 4.2: Ternary mental task classification by Bayesian quadratic classifier

Table 4.3 shows the results for quaternary combinations. In this table B/M/R/C refers to 4-nary classification of the four mental tasks of Baseline, Multiplication, Rotation and visual Counting.

Note that the number of 4-nary combinations for 5 tasks is 5, which can be calculated like $\binom{5}{4} = 5$, and for ternary and binary combination it is $\binom{5}{3} = \binom{5}{2} = 10$. This is the reason that for the 4-nary table there are 5 rows while for ternary and binary tables there are 10 rows. It is evident that for the five task combination, there is just one combination, B/M/R/C/L. The five task combination is evaluated in section 4.3. The results in Table 4.1-4.3 are further evaluated and explained in

Subjects	1	3	5	6
B/M/R/C	85.95 ± 4.57	**76.15** ± 4.41	71.59 ± 3.90	87.80 ± 4.23
B/M/R/L	86.55 ± 4.16	76.69 ± 4.41	73.04 ± 3.94	82.08 ± 4.65
B/M/L/C	86.30 ± 3.83	80.95 ± 4.42	**67.28** ± 4.68	79.78 ± 5.31
B/L/R/C	82.07 ± 5.07	78.23 ± 4.41	71.07 ± 4.02	**77.72** ± 5.52
L/M/R/C	83.16 ± 4.88	76.32 ± 5.14	73.80 ± 3.69	83.74 ± 4.80
Mean	84.80 ± 4.50	77.58 ± 4.46	71.36 ± 4.05	82.23 ± 4.90

Table 4.3: 4-nary mental task classification by Bayesian quadratic classifier

section 4.2.7.

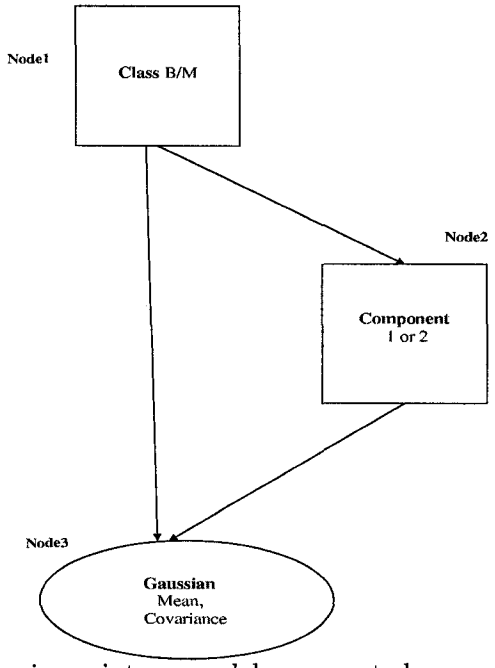

Figure 4.4: Gaussian mixture model represented as a graphical model.

4.2.3 Bayesian Network Classifier

As explained in 3.3, Bayesian networks provide a graphical representation of the variables in a problem and the relationships between them. Such a representation needs to be specified or learned from the data, and the structure together with the conditional density functions allows the multivariate density function to be specified through the product rule as in Equation 3.10.

The structure used for the Gaussian mixture is a Bayesian network. The graphical model corresponding to our Bayesian network is shown in Figure 4.4. Note that the square box in the figure corresponds to the input extracted features. The rectangular boxes correspond to the Gaussian mixture components. The rectangular boxes represent discrete values while the oval in the figure represents continuous values.

The graph structure of this model can be represented by the following adjacency matrix: 011 001 000. For every window of length one and overlap of half a second, the feature vectors were extracted. Features were AR coefficients of rank six, AR(6).

As mentioned earlier in section 3.7.1, a 5-fold cross validation is used to divide dataset between the training and the test sets. The classification was done 44 times for each individual combination of mental tasks. Next, the model is trained using the EM algorithm.EM works by starting with a randomly initialized model, and then iteratively refining the model parameters to produce a locally optimal maximum-likelihood fit.

The EM algorithm includes two steps. In the first step, each data point undergoes a soft-assignment to each mixture component. In the second step, the parameters of the model are adjusted to fit the data based on the soft-assignment of the previous step.[2]

The Bayesian Network Toolbox (BNT) of Matlab is used for implementing the classifier [6] and the results of four subjects can be seen in Table 4.4.

Subjects	1	3	5	6
B/M	93.43 ± 1.90	85.32 ± 2.48	79.89 ± 3.47	95.69 ± 2.06
B/C	97.82 ± 1.60	86.93 ± 3.53	73.23 ± 3.41	87.80 ± 5.03
B/L	93.81 ± 2.04	89.77 ± 3.08	75.67 ± 2.91	82.85 ± 4.25
B/R	95.17 ± 2.19	85.07 ± 3.16	88.07 ± 2.93	93.75 ± 1.74
M/C	95.23 ± 2.75	84.28 ± 4.43	79.46 ± 2.31	96.91 ± 1.58
M/L	96.62 ± 1.70	86.17 ± 4.88	79.75 ± 2.88	90.44 ± 2.61
M/R	95.90 ± 2.90	84.34 ± 3.64	88.49 ± 2.37	89.17 ± 2.44
C/R	87.06 ± 2.98	83.52 ± 3.69	87.64 ± 2.64	93.09 ± 1.95
C/L	90.75 ± 2.59	92.33 ± 3.43	81.78 ± 2.87	80.87 ± 3.41
L/R	92.33 ± 2.19	84.25 ± 5.66	90.85 ± 2.63	86.55 ± 2.96
Mean	93.81 ± 2.29	86.20 ± 3.08	82.48 ± 2.84	89.71 ± 2.80

Table 4.4: Classification results for Bayesian network classifier

[2]for more information on EM algorithm refer to [7]

4.2.4 Hidden Markov Models

As explained in section 3.4, HMM is a finite set of states, each of which is associated with a probability distribution that is generally multidimensional. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated according to the associated probability distribution.

In this thesis the observation is the extracted EEG features explained previously, which is generated by a Gaussian mixture model. This mixture model is characterized by three matrices for mean, variance and mixture percentages. There will be a transition matrix for moving between the two states. These parameters are all updated by the EM algorithm and at the end of training, there will be two trained HMMs corresponding to the two mental tasks [48].

To classify the test vectors given by the 5-fold cross validation scheme, the likelihood was calculated for them with respect to each of these HMMs. The HMM (mental task) that makes more likelihood is assigned to that vector. The results can be seen in Table 4.5.

Subjects	1	3	5	6
B/M	68.62 ± 11.6	53.16 ± 5.01	57.93 ± 7.87	69.25 ± 6.86
B/C	75.06 ± 8.56	58.40 ± 14.09	53.67 ± 5.34	66.11 ± 10.22
B/L	74.78 ± 8.31	74.68 ± 5.72	58.56 ± 10.18	62.07 ± 7.45
B/R	76.83 ± 8.52	57.54 ± 13.36	71.06 ± 6.78	76.70 ± 13.66
M/C	72.10 ± 4.44	62.85 ± 13.79	58.44 ± 6.51	56.22 ± 5.82
M/L	66.13 ± 7.78	68.37 ± 5.65	60.88 ± 8.58	61.96 ± 5.64
M/R	66.67 ± 6.48	56.75 ± 13.61	70.92 ± 7.56	67.64 ± 8.92
C/R	72.85 ± 7.11	62.41 ± 6.84	66.05 ± 7.54	62.12 ± 7.83
C/L	58.18 ± 7.28	79.67 ± 5.35	56.13 ± 7.70	55.33 ± 6.91
L/R	70.58 ± 8.57	67.11 ± 7.84	70.64 ± 10.25	68.69 ± 9.66
Mean	70.18 ± 7.82	64.10 ± 9.14	62.43 ± 7.83	64.61 ± 8.3

Table 4.5: Classification results for HMM classifier

4.2.5 Fisher Linear Classifier

As explained in section 3.5 there is a set of training patterns $x_1, x_2...x_N$, each of which is assigned to one of the two mental tasks, M_1 or M_2. Using this design set, weight vector $\mathbf{w}$ and a threshold $w0$ is defined such that:

$$\mathbf{w}^T\mathbf{x} + w0 \begin{cases} > 0 \Rightarrow \mathbf{x} \in M_1 \\ < 0 \Rightarrow \mathbf{x} \in M_2 \end{cases}$$

The Fisher approach would find a linear combination of the variables that separates the two classes as much as possible. That is, a direction is found, along which the two classes are best separated in some sense. The criterion proposed by Fisher is the ratio of between-class to within class variances as explained in section 3.5.

A Matlab function from the pattern recognition toolbox was used for this purpose [50]. This function computes the binary linear classifier based on the Fisher Linear Discriminant method.

The inputs are binary labeled training vectors. The parameter vector $\mathbf{w}$ of the linear classifier is computed to maximize the class separability criterion. The bias b is determined to lie between means of training data that is projected onto direction $\mathbf{w}$. The results can be seen in Table 4.6. Each entry is the average of 44 classification given by 5-fold cross validation algorithm.[3]

4.2.6 Neural Network

As was previously explained in section 3.6.1 in this section the classification results using feedforward neural network trained by a backpropagation algorithm will be explained[4]. These algorithms were previously developed on the Purdue dataset[4].

[3]refer to section 3.7.1 for more information
[4]refer to section ??

Subjects	1	3	5	6
B/M	91.22 ± 2.26	90.47 ± 4.06	77.35 ± 3.90	96.19 ± 3.52
B/C	92.83 ± 2.45	78.31 ± 4.58	73.86 ± 3.10	89.71 ± 3.75
B/L	89.61 ± 2.78	86.49 ± 4.28	80.27 ± 3.37	78.11 ± 4.68
B/R	95.45 ± 3.24	81.91 ± 4.75	88.07 ± 3.32	92.98 ± 2.31
M/C	95.90 ± 1.75	81.22 ± 3.39	73.88 ± 3.90	98.71 ± 1.28
M/L	95.45 ± 1.51	83.96 ± 4.01	78.33 ± 4.48	93.62 ± 2.36
M/R	95.99 ± 1.97	75.00 ± 5.20	87.17 ± 2.21	91.57 ± 2.60
C/R	85.26 ± 4.03	76.26 ± 3.69	85.55 ± 2.38	92.87 ± 3.37
C/L	80.93 ± 3.67	91.38 ± 2.44	81.15 ± 2.32	84.63 ± 4.20
L/R	88.86 ± 3.36	82.67 ± 4.29	92.27 ± 2.40	89.90 ± 3.27
Mean	91.15 ± 2.70	82.77 ± 4.13	81.79 ± 3.14	90.83 ± 3.11

Table 4.6: Classification results for Fisher linear classifier

In this section they are briefly introduced and the results are presented so further comparisons with other classifiers would be possible.

For this classifier a two layer feedforward neural network was implemented with 20 neurons in the hidden layer and one neuron in the output layer. A threshold of 0.5 was considered for the output neuron. Values greater than 0.5 and lower than 1 were assigned to one of the mental tasks and the values between 0 and 0.5 were assigned to the other task. Considering the 5-fold cross validation criteria, the classification was averaged over 44 times making use of different combinations of the training and test sets. The results can be seen in Table 4.7.

Another arrangement was also considered in the output with two neurons in the output layer, each corresponding to one mental task. For the Purdue dataset this arrangement gave us almost the same results as the one-neuron output [4]. This two-neuron structure was used for the Graz dataset and this will be explained later in section 4.3.

Subjects	1	3	5	6
B/M	91.67 ± 2.58	89.08 ± 3.37	79.26 ± 3.23	94.71 ± 2.89
B/C	96.59 ± 2.60	84.59 ± 4.40	76.68 ± 3.75	88.50 ± 4.05
B/L	93.34 ± 2.64	89.80 ± 3.02	78.59 ± 3.06	76.63 ± 4.84
B/R	95.01 ± 2.67	82.45 ± 4.75	87.60 ± 3.39	92.91 ± 2.91
M/C	95.42 ± 1.81	83.96 ± 4.29	76.66 ± 2.94	98.20 ± 1.68
M/L	95.61 ± 2.45	87.09 ± 4.75	79.68 ± 4.37	92.45 ± 2.87
M/R	97.03 ± 1.57	78.47 ± 4.54	87.68 ± 1.95	89.70 ± 2.28
C/R	86.90 ± 3.38	83.74 ± 4.83	86.26 ± 2.87	91.88 ± 3.63
C/L	88.79 ± 3.57	94.76 ± 2.81	80.45 ± 3.13	81.97 ± 3.36
L/R	90.81 ± 3.68	85.29 ± 3.68	91.25 ± 2.22	86.99 ± 3.26
Mean	93.12 ± 2.70	85.92 ± 4.05	82.41 ± 3.09	89.39 ± 3.18

Table 4.7: Classification results for neural network classifier

4.2.7 Comparison of Different Mental Tasks

So far, the results for different classifiers have been presented. In this section these results are compared and interesting conclusions will be reached, based on these comparisons.

The comparisons focus on each subject so that in Tables 4.8-4.11 the binary mental task results are separated for each individual subject. According to these tables, observations about optimal mental tasks for each subject will be explained.

Table 4.8 presents the classification results for subject six. It is obvious that in this table M/C[5] and B/M couples have the maximum values in binary combinations. Task M is common between these two binary pairs. It is interesting to note that in Table 4.3 the B/L/R/C combination, which does not have M, has the minimum in 4-nary combinations.

Table 4.2 shows that the B/L/C and B/L/R ternary combinations, which do not have M in them, have the minimum in 3-nary combinations. In these combinations,

[5]B,M,C,R and L refer to Baseline, Multiplication, Counting, Rotation and Letter writing mental tasks

the B/L pair is common. On the other hand, the B/L pair is the minimum in Table 4.8. So, it can be deduced that subject six's optimal mental task is M or Multiplication task and he is having problems with Baseline and Letter tasks. In Figure 4.10-4.12 there is a schematic diagram that clearly explains above statements.

For subject five there are three sessions of EEG signals, which is the most in the Purdue dataset compared to other subjects. Table 4.9 represents the results of classifiers for subject five. In this table the B/C pair has the minimum (except for neural network) while in Table 4.2 the L/M/R ternary combination has the maximum. These facts suggest that B/C is not a good pair of tasks for this subject.

Table 4.9 shows that the L/R, B/R, M/R and C/R pairs have the maximum values, all of which have the task R in common. On the other hand in Table 4.3 B/M/L/C has the minimum in 4-nary combinations. These facts suggest that it is the presence or absence of R that is making a difference for subject five. Thus, the Rotation task is proposed to be the optimal mental task for subject five.

Table 4.10 presents the classification results for subject three. This Table shows that the C/L pair has the maximum in all classifiers, while Table 4.2 shows that the B/M/R combination has the minimum in 3-nary tasks. This suggests that the C/L pair is the best for this subject. Table 4.3 shows that the B/M/R/C has the minimum in 4-nary combinations. The task L is not in this combination and can also be found in the C/L pair noted above. Thus, the L task or letter writing is the optimal mental task for subject three.

To summarize for subject six, task M was the optimal mental task and the B/L pair was the worst pair. For subject five the task R was the optimal mental task. For subject three the task L was the optimal mental task. For subject one such meaningful relations could not be found between mental tasks.

Method	Bayesian network	Neural network	Bayes Quadratic	Fisher
B/M	**95.78** ± 1.89	**94.71** ± 2.89	**97.23** ± 2.27	**96.19** ± 3.52
B/C	88.60 ± 4.37	88.50 ± 4.05	89.54 ± 4.55	89.71 ± 3.75
B/L	84.19 ± 4.44	76.63 ± 4.84	83.12 ± 4.93	78.11 ± 4.68
B/R	93.98 ± 1.09	92.91 ± 2.91	96.22 ± 2.42	92.98 ± 2.31
M/C	**97.19** ± 1.88	**98.20** ± 1.68	**98.83** ± 1.55	**98.71** ± 1.28
M/L	90.59 ± 2.67	92.45 ± 2.87	94.32 ± 2.48	93.62 ± 2.36
M/R	90.05 ± 3.22	89.70 ± 2.28	93.47 ± 3.20	91.57 ± 2.60
C/R	93.08 ± 1.62	91.88 ± 3.63	94.47 ± 2.08	92.87 ± 3.37
C/L	82.07 ± 3.83	81.97 ± 3.36	85.73 ± 5.02	84.63 ± 4.20
L/R	87.56 ± 2.37	86.99 ± 3.26	91.95 ± 3.76	89.90 ± 3.27
Mean	90.31 ± 2.77	89.39 ± 3.18	92.49 ± 3.23	90.83 ± 3.11

Table 4.8: Results of different classifiers for subject six

Method	Bayesian network	Neural network	Bayes Quadratic	Fisher
B/M	79.89 ± 3.47	79.26 ± 3.23	85.49 ± 3.42	77.35 ± 3.90
B/C	*73.23* ± 3.41	*76.68* ± 3.75	*79.81* ± 4.30	*73.86* ± 3.10
B/L	75.67 ± 2.91	78.59 ± 3.06	83.95 ± 4.24	80.27 ± 3.37
B/R	**88.07** ± 2.93	**87.60** ± 3.39	**89.06** ± 3.01	**88.07** ± 3.32
M/C	79.46 ± 2.31	76.66 ± 2.94	87.28 ± 3.43	73.88 ± 3.90
M/L	79.75 ± 2.88	79.68 ± 4.37	85.94 ± 3.28	78.33 ± 4.48
M/R	**88.49** ± 2.37	**87.68** ± 1.95	**88.55** ± 2.55	**87.17** ± 2.21
C/R	**87.64** ± 2.64	**86.26** ± 2.87	**88.07** ± 3.87	**85.55** ± 2.38
C/L	81.78 ± 2.87	80.45 ± 3.13	86.97 ± 3.52	81.15 ± 2.32
L/R	**90.85** ± 2.63	**91.25** ± 2.22	**90.66** ± 2.3	**92.27** ± 2.40
Mean	82.48 ± 2.84	82.41 ± 3.09	86.58 ± 3.40	81.79 ± 3.14

Table 4.9: Results of different classifiers for subject five

4.2.8 Preprocessing with ICA

As explained in section 3.7.5 ICA is a very general-purpose statistical technique in which randomly observed data are linearly transformed into components that are maximally independent from each other. A Matlab toolbox was used to implement the ICA which uses the infomax algorithm.

For the neural network and Bayesian quadratic classifier, all five mental tasks were

Method	Bayesian network	Neural network	Bayes Quadratic	Fisher
B/M	86.48 ± 3.27	88.95 ± 4.27	90.40 ± 3.07	90.47 ± 4.06
B/C	88.03 ± 3.01	82.67 ± 4.95	92.46 ± 3.03	78.31 ± 4.58
B/L	91.54 ± 3.12	89.33 ± 2.90	90.69 ± 2.93	86.49 ± 4.28
B/R	85.47 ± 5.13	82.82 ± 4.47	87.22 ± 4.22	81.91 ± 4.75
M/C	85.51 ± 3.93	83.61 ± 4.33	86.43 ± 4.00	81.22 ± 3.39
M/L	87.37 ± 4.71	85.76 ± 5.07	91.70 ± 3.99	83.96 ± 4.01
M/R	85.16 ± 3.72	77.36 ± 4.94	86.90 ± 3.61	75.00 ± 5.20
C/R	85.47 ± 5.09	81.84 ± 5.94	86.17 ± 4.71	76.26 ± 3.69
C/L	$\mathbf{93.71 \pm 3.06}$	$\mathbf{94.03 \pm 3.14}$	$\mathbf{94.44 \pm 2.64}$	$\mathbf{91.38 \pm 2.44}$
L/R	85.51 ± 4.77	84.05 ± 4.24	85.80 ± 3.68	82.67 ± 4.29
Mean	87.43 ± 3.98	85.04 ± 4.39	89.22 ± 3.56	82.77 ± 4.13

Table 4.10: Results of different classifiers for subject three

Method	Bayesian network	Neural network	Bayes Quadratic	Fisher
B/M	93.52 ± 1.84	90.53 ± 2.87	93.65 ± 2.19	91.22 ± 2.26
B/C	98.42 ± 1.60	95.35 ± 3.00	98.83 ± 1.04	92.83 ± 2.45
B/L	94.25 ± 2.08	91.76 ± 3.49	92.30 ± 3.12	89.61 ± 2.78
B/R	95.26 ± 1.99	95.42 ± 2.48	96.02 ± 2.93	95.45 ± 3.24
M/C	95.45 ± 2.63	95.80 ± 1.54	96.78 ± 1.98	95.90 ± 1.75
M/L	96.81 ± 1.62	95.83 ± 2.13	96.43 ± 2.48	95.45 ± 1.51
M/R	95.89 ± 2.99	96.96 ± 1.67	98.17 ± 1.61	95.99 ± 1.97
C/R	87.40 ± 2.86	85.66 ± 3.95	85.61 ± 5.39	85.26 ± 4.03
C/L	90.87 ± 2.32	87.75 ± 4.84	88.86 ± 4.72	80.93 ± 3.67
L/R	92.83 ± 2.57	89.74 ± 3.71	91.13 ± 3.28	88.86 ± 3.36
Mean	94.07 ± 2.25	92.48 ± 2.97	93.78 ± 2.87	91.15 ± 2.70

Table 4.11: Results of different classifiers for subject one

classified before and after the application of ICA. The results can be seen in Table 4.12.

The results in all subjects show a significant improvement because of the application of ICA. For the neural network the average of classification accuracy for all four subjects is 70.42% before the ICA application while the averaged classification accuracy is increased to 80.79% by using the ICA. Similarly, for the Bayesian quadratic

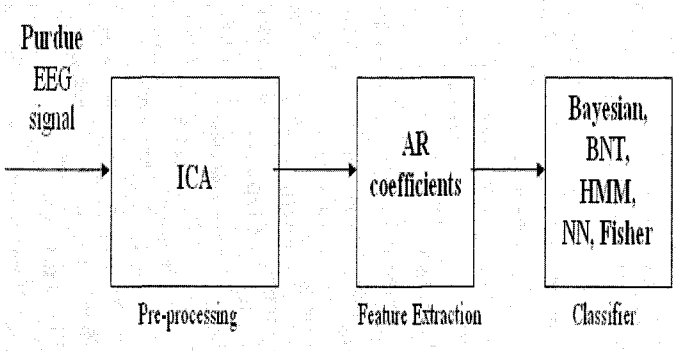

Figure 4.5: Block diagram of the Purdue dataset processing

classifier the averaged classification accuracy is 75.18% for all four subjects and this is increased to 86.56% by application of the ICA.

Altogether, at least a 10% increase in classification accuracy was achieved, which is significant. This shows the utility of ICA in artifact removal.

Subject	1	3	5	6
Neural Network	82.15 ± 3	68.66 ± 3.53	60.56 ± 3.35	70.33 ± 3.4
ICA+NN	90.33 ± 3.38	86.49 ± 3.1	65.34 ± 3.38	81.02 ± 3.51
Bayesian	82.64 ± 3.78	74.10 ± 3.9	65.57 ± 4.39	78.41 ± 4.34
Bayesian+ICA	92.82 ± 1.93	85.68 ± 4.7	81.04 ± 2.67	86.73 ± 3.1

Table 4.12: Results of ICA application

4.3 Graz Dataset Results

In the previous section, the Purdue dataset results were explained. In this section the results taken from the Graz dataset are evaluated. The main difference between the two datasets is that the Purdue dataset is taken during the performance of five mental tasks(B,M,R,L and C) while for the Graz dataset there are just two mental activities of left and right hand movement. On the other hand, the Graz dataset has

many more sessions compared to the Purdue dataset. In Figure 4.6 there is a block diagram of the algorithms that are applied to Graz dataset.

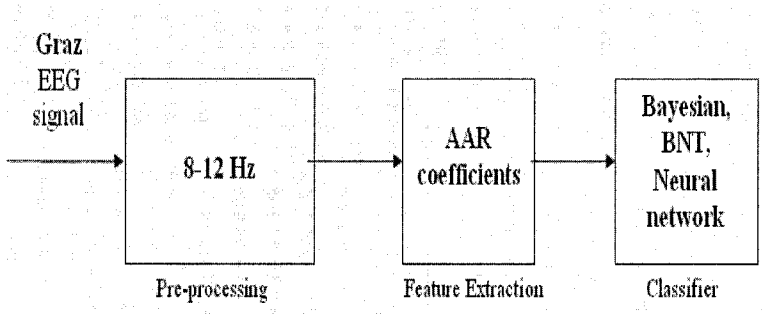

Figure 4.6: Block diagram of the algorithms applied to the Graz dataset

4.3.1 Graz Dataset

The Graz dataset was recorded from a normal female subject during a feedback session. The subject sat in a relaxing chair with armrests. The task was to control a feedback bar by means of imagery and left or right hand movements. The order of left and right cues were random.

The experiment consisted of 7 runs with 40 trials each. All runs were conducted on the same day with several minutes break in between them. At t=2s an acoustic stimulus indicated the beginning of the trial. The trigger channel went from low to high, and a cross '+' was displayed for 1 second. Then at t=3s, an arrow (left or right) was displayed as cue. At the same time the subject was asked to move a bar into the direction of the cue Figure 4.7.

The recording was made using a G.tec amplifier and Ag/AgCl electrodes [3]. Three bipolar EEG channels were measured over C3, Cz and C4. The EEG was sampled with 128Hz sampling rate and was filtered between 0.5 and 30Hz.

The trials for training and testing were randomly chosen. This can prevent any

systematic effect due to the feedback. Before discussing the results on Graz dataset first, the MI is calculated for this especial case. MI calculation facilitates the comparision of the results of the present study with results of other groups working on the same data.

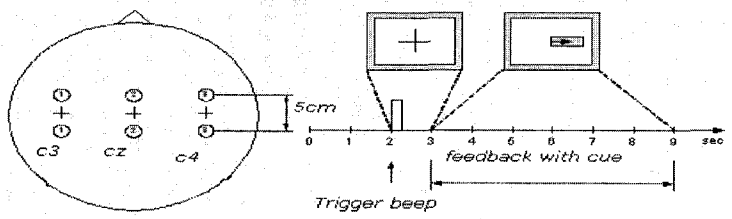

Figure 4.7: Electrode position(left) and timing scheme(right)

4.3.2 Mutual Information Calculation for Graz dataset

The concept of MI was previously explained in chapter three; the same concept is used here to calculate MI for the Graz dataset [44]. It is quite common to use the error rate for comparing different methods. However, the error rate takes into account just the sign of the classifier output but not the magnitude. For this reason, the MI is used to compare the different results. On the other hand, other groups have expressed their results on Graz dataset in the form of MI, so the present results can be compared with theirs by using MI.

If BCI output consists of two components one correlated with the desired output

and one uncorrelated:

$$o_k = u_k + e_k \tag{4.2}$$

where o_k is BCI's kth observed output, u_k is considered as useful signal, and e_k is for random noise. If the noise, e_k, is zero mean and variance, σ_e, and is not correlated with the signal:

$$e_k = N(0, \sigma_e^2) \tag{4.3}$$

$$E < (e_k - \mu_e)(u_k - \mu_u) >= 0 \tag{4.4}$$

In Graz BCI there are two possible output states. By considering the probabilities for both states to be equal, the mean and variance are $\mu_u = (\mu_1 + \mu_2)/2$ and $\sigma_u^2 = (\mu_1 - \mu_2)^2/4$ respectively

$$u_k = \{\mu_1, \mu_2\} \tag{4.5}$$

Accordingly, the signal-to-noise ratio (SNR) is

$$SNR = \sigma_u^2/\sigma_e^2 = \sigma_o^2/\sigma_e^2 - 1 \tag{4.6}$$

or in decibles:

$$SNR = 10 * log_{10}(\sigma_u^2/\sigma_e^2) \tag{4.7}$$

The MI, is the entropy of the useful output, i.e. the signal u, and is equal to the difference between the observed process o and the noise process e

$$MI(u) = H(o) - H(e) \tag{4.8}$$

If o and e are approximated as gaussians, from section 3.34:

$$MI(u) \approx 1/2log_2(2\pi e\sigma_o^2) - 1/2log_2(2\pi e\sigma_e^2) \tag{4.9}$$

$$MI(u) \approx 1/2log_2(\sigma_o^2/\sigma_e^2) = 1/2log_2(1 + SNR) \tag{4.10}$$

Equation above is an interesting equation, showing that the amount of information in the useful signal depends solely on SNR. In other words, the ratio between the signal and the noise variance determines the amount of MI between the output and the class relation. So, MI can be a good quantitative measure to assess BCIs and will be used in the next section to compare our results with other groups that are working on the Graz EEG dataset.

It is quite common to use the classification error for comparing different methods as in section 4.2. However, the classification error takes into account just the sign of the classifier output but not the magnitude. For this reason, the MI is used to compare the different results for the Graz dataset.

In Figure 4.8 the time course of error rate and MI is shown. The results of the Bayesian network classifier on the Graz dataset and will be explained in the next section.

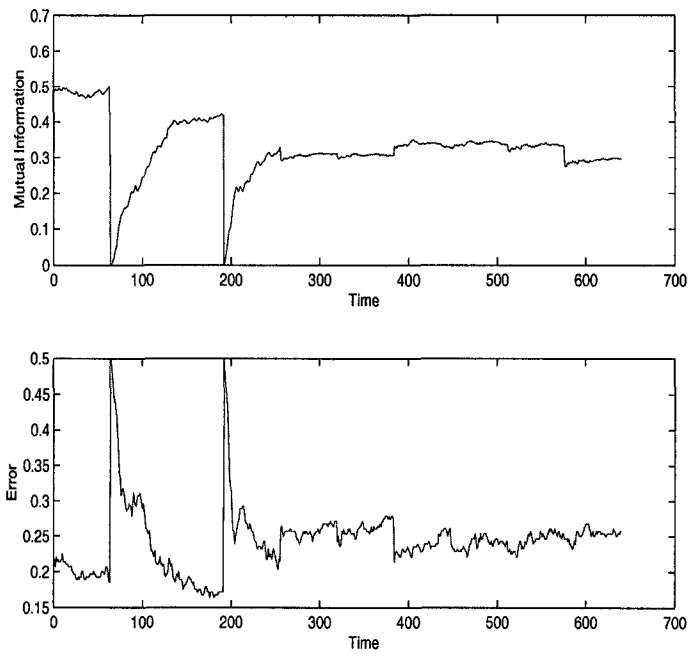

Figure 4.8: Time course of Mutual Information (bits) and error rate

4.3.3 Result Comparision

The BCI competition 2003 [3] was setup by four important BCI groups, Graz, Tubingen, Berlin and Wadsworth. Their works were explained in chapter two. This competition included different types of BCIs. In the present work, classification algorithms were applied to dataset III from the University of Graz. This dataset essentially includes the EEG signal while subjects imagine moving left or right hands.

Nine other groups have applied their algorithms to the same dataset and their results can be found in Table 4.13. The details of their work can also be found on the BCI 2003 website [3].

In the present study, as features, AAR coefficients of order six were extracted, leading to vectors of length twelve for two channels of EEG signal (i.e. C3 and C4). The Cz channel was discarded because it is in the middle of the scalp and is not associated with left or right movements. In the following three subsections the

Ranking	Groups	Minmum Error	Maximum SNR	Maximum MI
1	C	10.71	1.34	0.61
2	F	15.71	0.90	0.46
3	B	17.14	0.86	0.45
4	A	13.57	0.85	0.44
5	G	17.14	0.50	0.29
6	I	23.57	0.44	0.26
7	E	17.14	0.34	0.21
8	D	32.14	0.14	0.09
9	H	49.29	0.00	0.00

Table 4.13: Summary results of different groups in BCI competition 2003.

different classifiers will be explained.

Bayesian network implementation

Seven BNT's were trained on windows of length 1 second with an overlap of half a second. The test vectors results were averaged on corresponding overlapping windows classifiers. Originally the classifier gave two numbers corresponding to the probability of right or left. These two numbers were subtracted from each other and the result was considered as the classifier output. So, the final classifier output gives the difference of the probabilities of each vector belonging to class 1 or class 2.

So, the output should be positive for class 1, and negative for class 2, and zero for cases where the difference is lower than 1% threshold. This means that a reject option is also added to the classifier for cases where there is uncertainty about the results. This uncertainty is modeled in the form of a threshold value.

Different threshold values were considered but the results presented here are for a 1% threshold. For a threshold more than this there will be more rejected outputs or more zeros in the output.

Periods more than 3.5 seconds were considered for classification. The results for

the remaining five seconds can be seen in Figure 4.8. This figure shows the time course of MI and error rate for the BNT classifier.

The classifier was giving a positive number for a right hand signal and a negative number for a left hand and zero for non-decisive (reject) cases. Considering the 0.01 threshold for BNT, there were 3.70 % of the outputs to be zero or rejected. In other words, outputs between -0.01 and $+0.01$ were assigned to zero.

Neural network implementation

For the Graz dataset a neural network based approach was also considered for classification. For every trail (test or training), the signals were filtered using a bandpass 8-12Hz filter. The AAR coefficients of order six were then extracted for filtered and original C3 and C4 signals.

The neural network consisted of four layers. There were 20, 15 and 10 neurons in the first, second and third layers respectively and two neurons in the last layer, each corresponding to one class (i.e., right or left).

The neural network was trained on 2×11 different overlapping time regions of AAR coefficients of filtered and original signals of C3 and C4. Input time range was 128 samples (1 second) sliding 64 samples (0.5 second) for the next network. The neural network was trained for 300 epochs.

The test vectors were then applied to the trained neural networks and their classification results on overlapping regions were averaged. In non-overlapping time regions (first and last 0.5 s) the results of single networks were used. The final output of the classifier was considered to be the subtraction of output neurons values. Like the Bayesian network classifier, a reject option was also considered for the output whenever the output was less than 0.15 threshold.

In Figure 4.9 there is a sample output of the neural network and the classifier

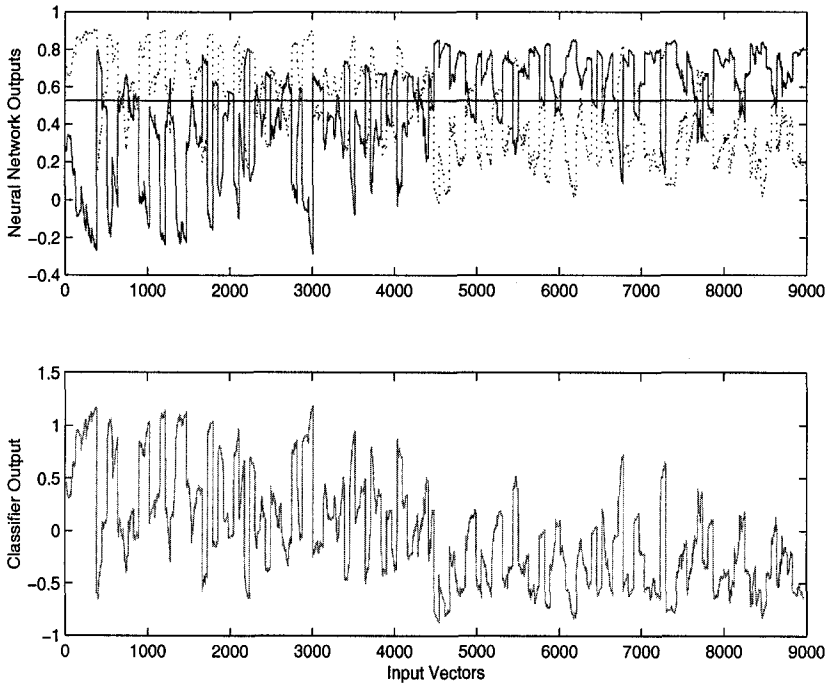

Figure 4.9: Neural network and classifier output for one window.

output. In the upper figure one of the output neurons is represented by dots and the other is represented in dash lines. The lower figure is the subtraction of the two curves above it and is considered as the final classifier output.

The first half of the vectors belong to class 1 and the second half belong to class 2. It can be verified that the output is mostly positive in the first half and negative in the second half. Those negative outputs in the first half are considered as misclassification and and those positive vectors in the second half are considered as misclassification as well.

Bayesian quadratic classifier implementation

For this classifier the same structure is used as in previous sections. The only difference was that the log-likelihood is calculated in this part that is explained in section 3.2.2.

Results

In general for the Graz dataset, three classifiers namely the Bayesian network, neural network and Bayesian quadratic classifier were used. The minimum of error, maximum of SNR, and maximum of MI were calculated for each classifier output. The results are shown in Table 4.14.

The comparison of Table 4.14 and Table 4.13 shows that by using the Bayesian network, the results from the current study rank second in maximum MI and maximum SNR and rank third in minimum error.

Classifier	Minmum Error	Maximum SNR	Maximum MI
Bayesian network	16.43	1.00	0.50
Neural network	15.71	1.04	0.51
Bayes classifier	17.14	0.71	0.38

Table 4.14: Application of three different classifiers to the Graz dataset

4.4 Summary

In this chapter the results of the application of algorithms explained in chapter three to the Purdue and Graz datasets were explained.

For the Purdue dataset it was found that for each of the subjects six, five and three a mental task can be assigned to be the optimal task, compared to the others. For subject five, the M task was the optimal mental task and the B/L pair was the optimal pair. For subject five the R task was the optimal mental task. For subject three the task L was the optimal mental task.

For subject one such relations could not be found among mental tasks. This might be justified by a higher percentage of classification accuracy for this subject in all tables compared to the others. This fact can especially be seen in Table 4.12

which is for all five mental tasks together.

Subject one might have been better trained in all tasks which has resulted in equally good results in all tasks so meaningful relations can not be found as were found for the other three subjects.

The ICA technique was applied to the Purdue dataset as the preprocessing block. All together there has been at least more than a 10% increase in classification accuracy, which is significant and shows the utility of ICA in artifact removal.

Similar good results were not obtained with the Graz dataset using the ICA. The reason is that for the Purdue dataset there is a separate channel for eye artifact. This seperate channel helps to separate this artifact as being an independent source using ICA. There is not such a direct representation of the eye artifact for Graz dataset.

For the Graz dataset, by calculating MI the results could be compared with other groups working on the same dataset. The results show that we rank second among the nine groups.

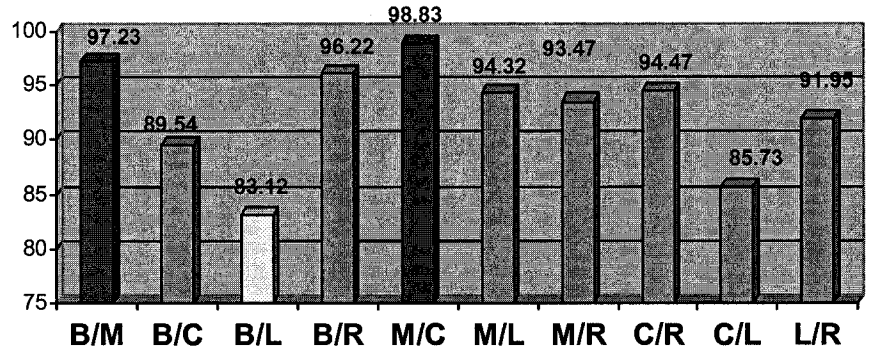

Figure 4-10 Binary classification of mental tasks for subject six. Correct classification versus different combinations.

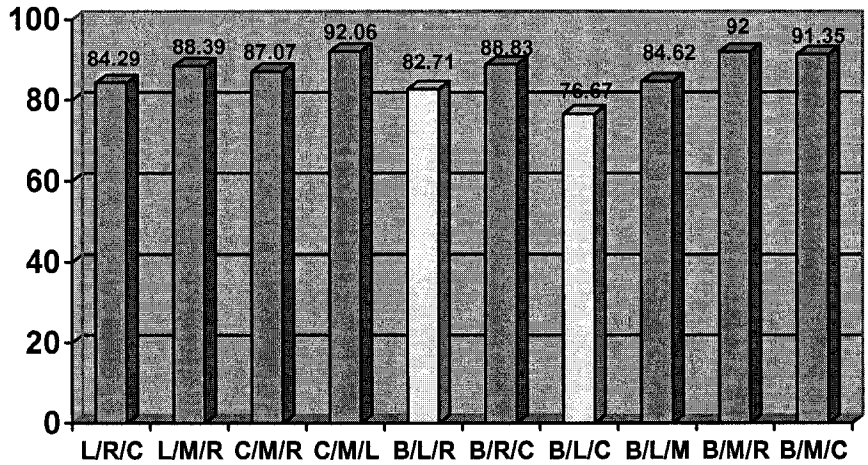

Figure 4-11 Ternary classification of mental tasks for subject six.

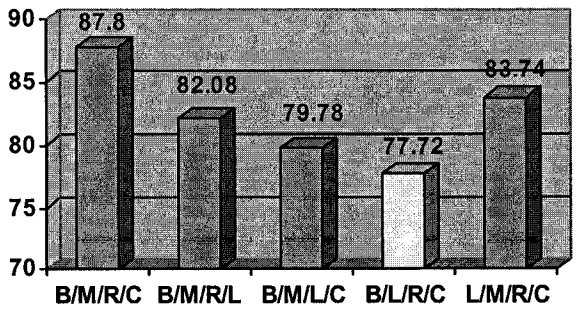

Figure 4-12 Quaternary (4-nary) classification of mental tasks for subject six.

Chapter 5

Conclusion

Introduction

In this chapter the results from the chapter four are discussed and the contributions of this research are explained. At the end of this chapter there will be an overview of future work that can be based on this research.

5.1 Contributions

In this research, the EEG signal was classified using different machine learning techniques. The algorithms were applied to two known datasets in the BCI field and the results were presented in the previous chapter.

For the first time in this work, the Bayesian network was used for EEG classification. This classifier was applied to both datasets, Purdue and Graz, which led to promising results compared to other techniques.

For the Purdue dataset and in all subjects, the classical Bayesian quadratic classifier has outperformed other classifiers. This classifier is not as good in the Graz dataset which could be expected. In the Graz dataset there are many more EEG

trials compared to the Purdue dataset and considering the non-stationary character of EEG, this leads to a less accurate estimation of mean and variance of the EEG signal, which is the base of this classifier.

Aside from the Bayesian quadratic classifier, and except for subject six, the Bayesian network is better than other classifiers. For subject one it is even better than the Bayesian quadratic classifier. This comparison is made according to classification accuracy. The Bayesian network continues to be as good in the Graz dataset. From the point of the standard deviation of the error, the Bayesian network has always been better compared to other classifiers and this means a more consistent classification. This improvement can also be seen in Figure 4.8 which gives an almost smooth curve of Mutual Information and classification error during time.

The HMM classifier had the poorest results compared to other classifiers. In the present study a simple structure of HMM was implemented. This might have been the reason for the HMM resulting in significantly poor results compared to other classifiers.

One of the most important challenges with the BCI systems is training. This training means that for most of the BCI systems based on mental tasks, subjects should go through several sessions of training. This training helps them to produce more distinguishable mental tasks. It is proven that untrained subjects are more comfortable with specific mental tasks compared to other tasks.

This important issue was addressed on the Purdue EEG dataset that has five mental tasks. Different combinations of these tasks were compared and for three of the subjects, one mental task was suggested to be superior to the others. This method can be a good approach for finding these tasks leading to a BCI with less amount of time spent on training.

One of the main problems with EEG analysis is the removal of artifacts. The

ICA technique had been used previously by other research groups for the removal of artifacts from the EEG signal [34]. In this research the ICA method was used for the same purpose. ICA has increased the classification accuracy by at least 10% in the Purdue dataset.

Similar good results were not obtained with the Graz dataset using the ICA. The reason is that for the Purdue dataset there is a separate channel for eye artifact. This helps to separate this artifact as being an independent source using the ICA. There is not such a direct representation of the eye artifact for the Graz dataset.

5.2 Future Work

As explained in the previous section different combinations of mental tasks were compared and for three of the subjects, one mental task was suggested to be the optimal mental task. In this research the Purdue dataset was used for this purpose but this dataset has been acquired in few sessions (at most three) while it is proven that EEG signal is highly non-stationary during the time. So, the acquired results on the Purdue dataset are just observations.

So, to be able to find a more general algorithm for finding optimal mental tasks there is a need for a larger multi-task, multi-session EEG dataset. By having such dataset one would be able to do statistical evaluation on results.

Such an algorithm can address two main problems of BCI systems, accuracy and ease of use. So, in future work more tasks can be given to each individual subject and then these tasks can be reduced to fewer tasks by the algorithm similar to section 4.2.7. Ideally, these fewer tasks are more easier for untrained subject to perform and results in better classification accuracy.

By the EEG instrument which exists now in UNBC and the robotic course presented here there is the possiblity for developing a real-time BCI and then connecting

this with robotic applications. This can be a very good example of BCI application to be able to move a robot by brain.

Considering different classifiers used in this thesis there might be a possibilty to merge all these in a hybrid classifier. This might improve the accuracy of classifications.

Bibliography

[1] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, P. H. Peckham, G. Schalk, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan. Brain-computer interface technology: A review of the first international meeting. IEEE Transactions on Rehabilitation Engineering, June 2000.

[2] Zachary A. Keirn Jorge I. Aunon "A New Mode of Communication between Man and His Surroundings" IEEE Trans. On BME, Vol. 37, No. 12, Dec 1990.

[3] Graz dataset: http://ida.first.fraunhofer.de/projects/bci/competition

[4] Kouhyar Tavakolian. Investigation and Comparison of Different Mental Task Classification by Linear and Nonlinear Techniques Applied to EEG Signal. Master of Science thesis, University of Tehran, July 2003.

[5] Janne Lehtonen. EEG-based Brain Computer Interfaces. Master of Science thesis, Helsinky University of Technology. May 2002

[6] Kouhyar Tavakolian, Siamak Rezaei, A.M.Nasrabadi, "Selecting Better EEG Channels for Classification of Mental Tasks" IEEE International Symposium On Circuits and Systems ISCAS2004. Volume III pages 537-540, Vancouver, Canada, May 2004.

[7] Todd K. Moon, Wynn C. Stirling. Mathematical Methods and Algorithms for signal processing. Prentice Hall 2000.

[8] Matlab software website: www.mathworks.com

[9] Bayesian Network Toolbox by Kevin Murphy: http://www.cs.ubc.ca/ murphyk/Software/BNT/bnt.html

[10] Michal Teplan, "Fundamentals of EEG measurement", Journal of the Institute of Measurement Science, Vol 2, Section 2.

[11] AnaesthesiaUK website: http://www.anaesthesiauk.com

[12] B.Y.Kim, K.S.Park "Automatic sleep stage scoring system using genetic algorithm and neural network" pro. 22nd IEEE, EMBS International Conference, Chicago USA, July 2000.

[13] http://www.brainwavescience.com/research.php

[14] Anon Cohen. Biomedical Signal Processing. CRC Press Inc 1983.

[15] Charles W. Anderson, Erik A. Stolz, and Sanyogita Shamsunder, "Multivariate Autoregressive Models for Classification of Spontaneous Electroencephalographic Signals During Mental Tasks" IEEE Transactions on Biomedical Engineering, Vol.45, No.3, March 1998.

[16] http://enterprise.aacc.cc.md.us/ rhs/bcipaper.html

[17] Jonathan R. Wolpaw, Niels Birbaumer, Dennis J. McFarland, Gert Pfurtscheller, and Theresa M. Vaughan. "Brain-Computer Interfaces for Communication and Control" Clinical Neurophysiology Vol.113,pp 767-791, 2002.

[18] Jonathan R. Wolpaw, Dennis J. McFarland, Theresa M. Vaughan, and Gerwin Schalk, "The Wadsworth Center Brain-Computer Interface (BCI) Research and Development Program" IEEE Transactions on Neural Systems and Rehabilitation Engineering, Vol. 11, No 2, pages 204-207,June 2003.

[19] Georg E. Fabiani, Dennis J. McFarland, Jonathan R. Wolpaw, and Gert Pfurtscheller "Conversion of EEG Activity into Cursor Movement by a Brain-Computer Interface (BCI)" IEEE Transactions on Neural Systems and Rehabilitation Engineering, Vol. 12, No 3, September 2004.

[20] Dennis J. McFarland, William A. Sarnacki, Theresa M. Vaughan, Jonathan R. Wolpaw "Brain-computer interface (BCI) operation: signal and noise during early training sessions" Clinical Neurophysiology Vol. 116, pages 56-62, Jan 2005.

[21] G. Pfurtscheller, C. Neuper, A. Schlgl, and K. Lugger, "Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters" IEEE Trans. Rehab. Eng. vol. 6, pp. 316-325, 1998.

[22] C. Guger, A. Schlgl, C. Neuper, D. Walterspacher, T. Strein, and G. Pfurtscheller, "Rapid Prototyping of an EEG-based brain-computer interface (BCI)" IEEE Trans. Rehabilitation Eng, vol. 9, pages 1-10, 2001.

[23] Niels Birbaumer, Andrea Kbler, Nimr Ghanayim, Thilo Hinterberger, Jouri Perelmouter, Jochen Kaiser, Iver Iversen, Boris Kotchoubey, Nicola Neumann, and Herta Flor "The Thought Translation Device (TTD) for Completely Paralyzed Patients" IEEE Transaction on Rehabilitation Engineering, Vol. 8, No. 2, June 2000.

[24] T. Hinterberger, A. Kbler, J. Kaiser, N. Neumann, and N. Birbaumer, "A brain-computer-interface (BCI) for the locked-in: Comparison of different EEG classifications for the thought translation device (TTD)" Clin. Neurophysiol., vol. 114, pages 416-425, 2003.

[25] Thilo Hinterberger, Stefan Schmidt, Nicola Neumann, Jrgen Mellinger, Benjamin Blankertz, Gabriel Curio and Niels Birbaumer "Brain-Computer Communication and Slow Cortical Potentials" IEEE Transaction on Biomedical Engineering, Vol.51, No.6, June 2004.

[26] Guido Dornhege, Benjamin Blankertz, Gabriel Curio, and Klaus-Robert Mller "Boosting bit rates in non-invasive EEG single-trial classifications by feature combination and multi-class paradigms" IEEE Trans. Biomed. Eng., Vol 51, pages 993-1002, 2004.

[27] http://www.ece.ubc.ca/ garyb/BCI.htm

[28] Birch, G.E., Bozorgzadeh, Z., and Mason, S.G., "Initial On-Line Evaluations of the LF-ASD Brain-Computer Interface with Able-Bodied and Spinal-Cord Subjects using Imagined Voluntary Motor Potentials" Proceedings of the 4th international ACM conference on assistive technologies, Arlington, pages 109-113.

[29] K.-R. Mueller, C. Anderson and G.E. Birch "Linear and Nonlinear Methods for Brain-Computer Interfaces" submitted to IEEE Trans. Neural Sys. and Rehab. Engineering, Vol. 11, No. 2,pages 165-169, Jan 2003.

[30] Adaptive Brain Interface project: http://sir.jrc.it/abi/

[31] Milln J. del R., Mourio J., Cincotti F., Babiloni F., Varsta M., and Heikkonen J. "Local neural classifier for EEG-based recognition of mental tasks" IEEES International Joint Conference on Neural Networks. Como, Italy 2000.

[32] Jos del R. Milln, Frdric Renkens, Josep Mourio, Wulfram Gerstner "Brain-actuated interaction" Artificial Intelligence. Vol. 159, pages 241-259, Nov 2004.

[33] Gary Nelson Garcia Molina. Direct Brain-Computer Communication through Scalp Recorded EEG Signals. Ecole Polytechnique De Lausanne. Ph.D. thesis 2004.

[34] M. Ungureanu, C. Bigan, R. Strungaru, V. Lazarescu, "Independent Component Analysis Applied in Biomedical Signal Processing" Measurement Science Review, Volume 4, Section 2, 2004

[35] Purdue dataset: http://www.cs.colostate.edu/eeg/?Summary

[36] Kouhyar Tavakolian, Siamak Rezaei, A.M.Nasrabadi "Classification of Different Mental Tasks Using Neural Network" WSEAS Transactions on Electronics pages: 343-347, Issue2, Volume1, April 2004.

[37] Kouhyar Tavakolian, Siamak Rezaei "Genetic Algorithm for Feature Reduction in Brain Computer Interfaces" IASTED International Conference on Biomedical Engineering. Pages 289-292. Innsbruck, Austria. Feb 2004.

[38] Kouhyar Tavakolian, A.M.NasrAbadi, S.K Setarehdan , M.A Khalilzadeh, "Effects of Different Feature Vectors and Neural Network Topology on EEG Mental Task Classification" Proceedings of IASTED International Conference on Biomedical Engineering, Biomed2003, Salzburg, Austria, pages 39-43, June 2003.

[39] Andrew Webb. Statistical pattern recognition. Oxford University Press, 1999.

[40] John G. Webster. Medical Instrumentation Application and Design. Wiley 1998.

[41] Khalid Sayood. Data Compression. Morgan Kaufman Publisher 2000.

[42] A. Schloegl. The electroencephalogram and the adaptive autoregressive model. Shaker Verlag 2000.

[43] A. Schloegl, K. Lugger and G. Pfurtscheller. "Using Adaptive Autoregressive Parameters for a Brain-Computer-Interface Experiment". Proceedings of 19th International IEEE EMBS conference, pages 1533-1535, Chicago 1997.

[44] A. Schlgl, C. Keinrath, R. Scherer, G. Pfurtscheller "Information transfer of an EEG-based Bran-computer interface". Proceedings of the 1st International IEEE EMBS Conference on Neural Engineering, Capri, Italy, March 2003.

[45] Aapo Hyvarinen, Erkki Oja, "Independent Component Analysis" Neural Networks, Vol.13, pages 411-430, 2000.

[46] http://www.sccn.ucsd.edu/eeglab/

[47] L. Rabiner "A tutorial on Hidden Markov Models and selected applications in speech recognition" Proc. IEEE Vol 77, pages 257-286, 1989.

[48] Shi Zhong and Joydeep Ghosh. "HMMs and coupled HMMs for multi-channel EEG classification". Proc IEEE International Joint Conference on Neural Networks, May 2002.

[49] Finn V. Jensen. Bayesian Networks and Decision Graphs. Springer 2001.

[50] http://cmp.felk.cvut.cz/ xfrancv/stprtool/

[51] Simon Haykin. Neural Networks. Prentice Hall 1999.

[52] A. Papoulis, S. Unnikrishna Pillai. Probability, Random Variables and Stochastic Processes. McGraw-Hill December, 2001

Appendix A

Program Codes

In this section some important Matlab codes used for the implementation of algorithms are presented.

A.1 Feature extraction

A.1.1 Feature extraction for Graz dataset

```
g=size(datach); %datach includes EEG data
SiDwin=g(1,1)/128;%128 HZ is the sampling frequency
for trial=1:70,
for ij=0:SiDwin-1,
    for ch=1:3,
        st=datach(1+ij*128:128+ij*128,ch,trial);
        [aar1,e,REV] = aar(st-mean(st),[1,2],[6]);
        pfar(6*(ch-1)+1:6*(ch-1)+6,128*ij+1:128*ij+128,trial)=aar1';
    end
end
end
```

A.1.2 Feature extraction for Pudue dataset

```
for ij=0:(SiDwin-2),
    for j=1:6,
        st=datach(j,(1+ij*125:250+ij*125))'; %250Hz is the sampling frequency
        model=ar(st-mean(st),6,'burg');
        arc=th2poly(model);
        arc=arc(1,2:7);
        pfar(6*(j-1)+1:6*(j-1)+6,ij+1)=arc';
    end
end
```

A.2 Bayesian network for Graz dataset

```
load Kgrazfeature.mat %this file includes the extracted AAR features
load KgrazKiaadd.mat
%loading left hand extracted features
ML1=MatL(:,1:128,:); %128 is equal to one second of Graz EEG dataset

MR1=MatR(:,1:128,:);%loading right hand exteacted features

dag = [ 0 1 1 ; 0 0 1 ; 0 0 0 ]; %the DAG adjacancy matrix
discrete_nodes = [1 2];
nodes = [1 : 3];
node_sizes=[ 2 2 12];

M=MR1;
N=ML1;
p=1;
for i=1:70,
   for j=1:128,
       A(:,p)=M(:,j,i);
       B(:,p)=N(:,j,i);
      p=p+1;
   end
end

nnw=1
Ktrain=[A B];
koh1=length(Ktrain)/2;

jjjj=0;
koh1=length(Ktrain)/2;
bnet = mk_bnet(dag, node_sizes, 'discrete', discrete_nodes);
bnet.CPD{1} = tabular_CPD(bnet,1);
bnet.CPD{2} = tabular_CPD(bnet,2);
bnet.CPD{3} = gaussian_CPD(bnet, 3);
ktrain=Ktrain';
trainingX = ktrain;
[Tr1 hhh]=size(ktrain);
Tr1=Tr1/2;
trainingC(1:Tr1) = 1;   %% Class 1 is right hand
trainingC(Tr1+1:2*Tr1) = 2; %% Class 2 is left hand
```

```
training= cell(3,length(trainingX));
training(3,:) = num2cell(trainingX',1);
training(1,:) = num2cell(trainingC,1);
engine = jtree_inf_engine(bnet);
maxiter=4;      %% The number of iterations of EM
epsilon=1e-100; %% A small stopping criterion
[bnet1B, ll, engine1B] = learn_params_em(engine,training,maxiter,epsilon);
class0= cell(3,1); %% Create an empty cell array for observations
class1 = class0;
class2 = class0;
class1{1} = 1;      %% The class node is observed to be right hand
class2{1} = 2;      %% The class node is observed to be left hand
for i=1:Tr1
   sample1=sample_bnet(bnet1B,'evidence',class1);
   sample2=sample_bnet(bnet1B,'evidence',class2);
   modelX(i,:)=sample1{3}';
   modelX(i+Tr1,:)=sample2{3}';
end

% Testing the Bayesian network classifier
M=Rtest1; %loading test vectors
N=Ltest1;
p=1;
for i=1:70,
   for j=1:64,
       A(:,p)=M(:,j,i);
       B(:,p)=N(:,j,i);
       p=p+1;
   end
end

clear p;
ktest=[A B];
ktestN=ktest';
testX = ktestN;
[Te1 kkk]=size(ktestN);
evidence=class0;   %% Start out with nothing observed
for i=1:Te1
   evidence{3}=testX(i,:)';
   [engine1BB, ll] = enter_evidence(engine1B,evidence);
   marg = marginal_nodes(engine1BB,1);
   pB1(i,:)=marg.T';
end
```

```
[gf,bf]=size(pB1);
ZX1=pB1';
Out=ZX1(1,:)-ZX1(2,:);
```

A.3 Neural network for Graz dataset

```
load Kgrazfeature.mat %this file includes the extracted AAR features
load KgrazKiaadd.mat
%loading left hand extracted features
ML1=MatL(:,1:128,:); %128 is equal to one second of Graz EEG dataset
%loading right hand exteacted features
MR1=MatR(:,1:128,:);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
M=MR1;
N=ML1;
p=1;
for i=1:70,
   for j=1:128,
       A(:,p)=M(:,j,i);
       B(:,p)=N(:,j,i);
      p=p+1;
   end
end
nnw=1
A=[A MatRFadd(:,1:2920)];
B=[B MatLFadd(:,1:2920)];
Ktrain=[A B];
koh1=length(Ktrain)/2;
koh1=length(Ktrain)/2;
tar= [ones(1,koh1)    0*ones(1,koh1);
      0*ones(1,koh1)   ones(1,koh1)]; %target vectors
net1Z=newff(([min(Ktrain'); max(Ktrain')])',[30,2],{'tansig' 'tansig'});
%Neural network architecture
net1Z.trainFcn ='traingdx';
net1Z.trainParam.epochs=200;  %number of epochs
net1Z.trainParam.show=NaN;
[net1Z,tr]=train(net1Z,Ktrain,tar);%training the neural network
Win=1;
errB=0;
M=Rtest1;
N=Ltest1;
p=1;
```

```
for i=1:70,
   for j=1:64,
       A(:,p)=M(:,j,i);
       B(:,p)=N(:,j,i);
       p=p+1;
   end
end
ktest=[A B];
YtZO=sim(net1Z,ktest); %application of test vectors to the trained neural netwo
Out=YtZO(1,:)-YtZO(2,:);%final output of the classifier
```

A.4 Bayesian quadratic classifier for Graz dataset

```
load Kgrazfeature.mat %this file includes the extracted AAR features
load KgrazKiaadd.mat
%loading left hand extracted features
ML1=MatL(:,1:128,:); %128 is equal to one second of Graz EEG dataset
%loading right hand exteacted features
MR1=MatR(:,1:128,:);
M=MR1;
N=ML1;
p=1;
for i=1:70,
   for j=1:128,
       A(:,p)=M(:,j,i);
       B(:,p)=N(:,j,i);
      p=p+1;
   end
end
Ktrain=[A B];
muR_1=mean(A');
sigmaR_1=cov(A');
muL_1=mean(B');
sigmaL_1=cov(B');
%testing
M=Rtest1;
N=Ltest1;
p=1;
for i=1:70,
   for j=1:64,
       A(:,p)=M(:,j,i);
       B(:,p)=N(:,j,i);
       p=p+1;
```

```
   end
end
clear p;
ktest=[A B]';
Tel=length(ktest);
for i=1:Tel
logLike1 = (ktest(i,:) - muR_1) * inv(sigmaR_1) * (ktest(i,:)- muR_1)';
logLike1 = - 0.5 * (logLike1 + log(det(sigmaR_1)));
p(i,1)=logLike1;
end
for i=1:Tel
logLike2 = (ktest(i,:) - muL_1) * inv(sigmaL_1) * (ktest(i,:)- muL_1)';
logLike2 = - 0.5 * (logLike2 + log(det(sigmaL_1)));
p(i,2)=logLike2;
end
pB1=p;
```

A.5 Fisher linear classifier for Graz dataset

```
for u=1:200,
load Kgrazfeature.mat
Ktrain=[MatR MatL];
data.X=Ktrain;
data.y=[1*ones(1,length(Ktrain)/2) 2*ones(1,length(Ktrain)/2)];
model = fld(data)
for trial=1:70
MatRtest=Matrtest(:,:,trial);
MatLtest=Matltest(:,:,trial);
ktest=[MatRtest MatLtest];
tst.X=ktest;
ypred(trial,:)=linclass(tst.X,model);
tst.y=[1*ones(1,length(ktest)/2) 2*ones(1,length(ktest)/2)];
ccc(trial)=100*(1-cerror(ypred(trial,:),tst.y))
end
aa(u)=mean(ccc)
end
```

A.6 HMM for Graz dataset

```
load labels_data_set_iii.mat;
load KgrazICA0.mat;
datach=dataLTEST;
```

```
featureoctaar; %feature extraction
Matltest=pfar;
datach=dataRTEST;
featureoctaar;
Matrtest=pfar;



datach=dataLTR;
featureoct2; %feature extraction
Matl=pfar;

datach=dataRTR;
featureoct2;
Matr=pfar;
p=1;
for i=1:70,
   for j=1:23,
       MatR(:,p)=Matr(:,i,j);
       MatL(:,p)=Matl(:,i,j);
              p=p+1;
   end
end
%[pp,q]=size(Matl);
O = 3;
M = 1;
Q = 2; Number of HMM states
left_right = 0;
Ktrain=[MatR MatL];
koh1=length(Ktrain)/2;
ktest=[Matrtest Matltest];
koh2=length(ktest)/2;
Ktrainmat1=[MatR];
Ktrainmat2=[MatL];
K1=reshape(MatR,3,6,koh1);
data=K1;
prior0 = normalise(rand(Q,1));
transmat0 = mk_stochastic(rand(Q,Q));
cov_type='full';
method='kmeans';
[mu0, Sigma0] = mixgauss_init(Q*M,data, cov_type,method);
mu0 = reshape(mu0, [O Q M]);
```

```
Sigma0 = reshape(Sigma0, [O O Q M]);
mixmat0 = mk_stochastic(rand(Q,M));
[LL, prior1, transmat1, mu1, Sigma1, mixmat1] = ...
    mhmm_em(data, prior0, transmat0, mu0, Sigma0, mixmat0, 'max_iter', 10);
clear data
K2=reshape(MatL,3,6,koh1);
data=K2;
prior2 = normalise(rand(Q,1));
transmat2 = mk_stochastic(rand(Q,Q));
cov_type='full';
method='kmeans';
[mu2, Sigma2] = mixgauss_init(Q*M,data, cov_type,method);
mu2 = reshape(mu2, [O Q M]);
Sigma2 = reshape(Sigma2, [O O Q M]);
mixmat2 = mk_stochastic(rand(Q,M));
[LL, prior3, transmat3, mu3, Sigma3, mixmat3] = ...
    mhmm_em(data, prior2, transmat2, mu2, Sigma2, mixmat2, 'max_iter', 10);
clear data;

Ktest1=reshape(Matrtest,3,6,koh2);
for ii=1:koh2,
data=Ktest1(:,:,ii);
[loglik, errors] = mhmm_logprob(data, prior1, transmat1, mu1, Sigma1, mixmat1);
p(ii,1)=loglik;
[loglik, errors] = mhmm_logprob(data, prior3, transmat3, mu3, Sigma3, mixmat3);
p(ii,2)=loglik;
end


Ktest2=reshape(Matltest,3,6,koh2);
for ii=1:koh2,
data=Ktest2(:,:,ii);
[loglik, errors] = mhmm_logprob(data, prior1, transmat1, mu1, Sigma1, mixmat1);
p(ii+koh2,1)=loglik;
[loglik, errors] = mhmm_logprob(data, prior3, transmat3, mu3, Sigma3, mixmat3);
p(ii+koh2,2)=loglik;
end

[gf,bf]=size(p);

err=0;
for mm=1:gf/2
    if p(mm,1)<p(mm,2)
```

```
        err=err+1;
    end
end

for nn=gf/2+1:gf
    if p(nn,1)>p(nn,2)
        err=err+1;
    end
end

Percentcorrect=100*(1-err/gf)
```

A.7 Bayesian network classifier for Purdue dataset

```
SiD1=fix(length(datachmat2)/125);
SiD2=fix(length(datachmat1)/125);
SiDwin=min(SiD1,SiD2);
SiD=125*SiDwin;
datachmat2=datachmat2(:,1:SiD);
datachmat1=datachmat1(:,1:SiD);
datach=datachmat2;
featuremay; %feature extraction
Mat2=pfar;
datach=datachmat1;
featuremay;
Mat1=pfar;

koh=fix(length(Mat2)/10)*10;
SiDwin=SiDwin-1;
ResMat1=Mat1(:,koh+1:SiDwin);
Mat1=Mat1(:,1:koh);
ResMat2=Mat2(:,koh+1:SiDwin);
Mat2=Mat2(:,1:koh);
[pp,q]=size(Mat1);

load xvalid.mat
dag = [ 0 1 1 ; 0 0 1 ; 0 0 0 ];
discrete_nodes = [1 2];
nodes = [1 : 3];
node_sizes=[ 2 2 36]; %36 is the size of EEG extracted features
jjjj=0;
for r=1:44 %5-fold cross validation different permutations
```

```
x=xvalidmat(r,:);
Ktrainmat1=Mat1(:,[x(3):10:q x(4):10:q x(5):10:q x(6):10:q x(7):10:q
x(8):10:q x(9):10:q x(10):10:q]);
ktestmat1=Mat1(:,[x(1):10:q  x(2):10:q]) ;
Ktrainmat2=Mat2(:,[x(3):10:q x(4):10:q x(5):10:q x(6):10:q x(7):10:q
x(8):10:q x(9):10:q x(10):10:q]);
ktestmat2=Mat2(:,[x(1):10:q x(2):10:q]) ;
Ktrain=[Ktrainmat2 ResMat2 Ktrainmat1 ResMat1];
koh1=length(Ktrain)/2;
ktest=[ktestmat2 ktestmat1];
bnet = mk_bnet(dag, node_sizes, 'discrete', discrete_nodes);
bnet.CPD{1} = tabular_CPD(bnet,1);
bnet.CPD{2} = tabular_CPD(bnet,2);
bnet.CPD{3} = gaussian_CPD(bnet, 3);

ktrain=Ktrain';
trainingX = ktrain;
[Tr1 hhh]=size(ktrain);
Tr1=Tr1/2;
ktestN=ktest';
testX = ktestN;
[Te1 kkk]=size(ktestN);
 p=zeros(Te1,2);

trainingC(1:Tr1) = 1;   %% Class 1 is Right hand
trainingC(Tr1+1:2*Tr1) = 2; %% Class 2 is Left hand
training= cell(3,length(trainingX));
training(3,:) = num2cell(trainingX',1);
training(1,:) = num2cell(trainingC,1);
engine = jtree_inf_engine(bnet);
maxiter=6;     %% The number of iterations of EM (max)
epsilon=1e-100; %% A very small stopping criterion
[bnet2, ll, engine2] = learn_params_em(engine,training,maxiter,epsilon);
class0= cell(3,1); %% Create an empty cell array for observations
class1 = class0;
class2 = class0;
class1{1} = 1;     %% The class node is observed to be Right hand
class2{1} = 2;     %% The class node is observed to be Left hand

for i=1:Tr1
   sample1=sample_bnet(bnet2,'evidence',class1);
   sample2=sample_bnet(bnet2,'evidence',class2);
   modelX(i,:)=sample1{3}';
```

```
   modelX(i+Tr1,:)=sample2{3}';
end
evidence=class0;   %% Start out with nothing observed
for i=1:Te1
   evidence{3}=testX(i,:)';
   [engine3, ll] = enter_evidence(engine2,evidence);
   marg = marginal_nodes(engine3,1);
   p(i,:)=marg.T';
end

[gf,bf]=size(p);

err=0;
for mm=1:gf/2
    if p(mm,1)<p(mm,2)
        err=err+1;
    end
end

for nn=gf/2+1:gf
    if p(nn,1)>p(nn,2)
        err=err+1;
    end
end

Percentcorrect=100*(1-err/gf)
xvalidresults(1,r)=Percentcorrect;
end
payan(1,1)=mean(xvalidresults);
payan(1,2)=std(xvalidresults);
```

Appendix B

Related Published Papers

These papers are published during this research work. They are based or are related on the research works that were explained in this thesis:

[1] Siamak Rezaei, Kouhyar Tavakolian, "Comparison of Five Different Classifiers for Classification of Mental Tasks", 27th International IEEE-EMBC, Shanghai China, Sept 2005.

[2] Kouhyar Tavakolian, Siamak Rezaei, "Classification of Mental Tasks Using Gaussian Mixture Bayesian Network Classifiers" IEEE international workshop on biomedical circuits and systems. Singapore. Dec 2004.

[3] Kouhyar Tavakolian, Siamak Rezaei, S.K.Starehdan, "Choosing Optimal Mental Tasks for Classification in Brain Computer Interfaces" IASTED Conference on Artificial Intelligence and Applications. Pages 396-399. Innsbruck. Feb 2004.

[4] Kouhyar Tavakolian, Siamak Rezaei, "Mental Task Classification Using Different Classifiers" accepted for publication in WSEAS conference in Systems, Vouliagmeni, Greece, July 2005.

[5] Kouhyar Tavakolian, Siamak Rezaei, A.M.Nasrabadi, "Selecting Better EEG Channels for Classification of Mental Tasks" IEEE International Symposium On Circuits and Systems ISCAS2004. VolumeIII pages 537-540. Vancouver. May 2004.

[6] Kouhyar Tavakolian, Siamak Rezaei, A.M.Nasrabadi "Classification of Different Mental Tasks Using Neural Network" WSEAS Transactions on Electronics pages: 343-347, Issue2, Volume1, April 2004.

[7] Kouhyar Tavakolian, Siamak Rezaei, "Genetic Algorithm for Feature Reduction in Brain Computer Interfaces" IASTED, International Conference on Biomedical Engineering. Pages 289-292. Innsbruck, Austria. Feb 2004.

[8] Saeedeh Parsaeian, Kouhyar Tavakolian, M.B.shamsolahi, Siamak Rezaei "Classification of Knee Joint Vibroarthrographic Signals Using Feedforward Neural Network" IASTED International Conference on Biomedical Engineering. Pages 343-345. Innsbruck, Austria. Feb 2004

[9] Siamak Rezaei Kouhyar Tavakolian, "Planning for BioMedical Computing" The Western Canadian Conference on Computing Education, Prince George, Canada, May 2005.